CRISP-DM no Desenvolvimento de Funções de Pedotransferência: Um Estudo de Caso com o Banco de Dados HYBRAS

CRISP-DM in the Development of Pedotransfer Functions: A Case Study with the HYBRAS Database

Estevão Lucas Ramos da Silva[1], **Antônio Correia de Sá Barreto Neto**[1]

[1]Análise e Desenvolvimento de Sistemas – Instituto Federal de Pernambuco - (IFPE)
Caixa Postal 53441-601 – Maranguape 1 – Paulista – Brazil

hurryblank@gmail.com, antonio.neto@paulista.ifpe.edu.br

Resumo. *Esta pesquisa descreve uma análise detalhada sobre a criação de funções de pedotransferência para estimar a Capacidade de Campo e o Ponto de Murcha Permanente. A abordagem adotada utiliza dados provenientes do banco HYBRAS que detém informações de constantes hidráulicas do solo, e segue a metodologia CRISP-DM, no qual oferece uma estrutura padronizada para o desenvolvimento de modelos. O estudo envolve a construção de doze modelos de inteligência artificial, explorando algoritmos que buscam relações tanto lineares quanto não lineares. O algoritmo de Gradient Boosting demonstrou o melhor desempenho para estimar o Ponto de Murcha Permanente, alcançando um coeficiente de determinação (R^2) de 0.74 e um Erro Quadrático Médio (RMSE) de 0.04 cm^3/cm^3. O projeto destaca a intenção de dar protagonismo aos especialistas durante o desenvolvimento das funções, ressaltando a relevância da participação ativa desses profissionais ao longo de todas as etapas do processo.*
Palavras-chave: CRISP-DM, Hybras, Funções de Pedotransferência, Umidade do solo e Mineração de Dados

Abstract. *This research describes a detailed analysis on the development of pedotransfer functions to estimate Field Capacity and Permanent Wilting Point. The adopted approach utilizes data from the HYBRAS database, which holds information on soil hydraulic constants, and follows the CRISP-DM methodology, providing a standardized framework for model development. The study involves the construction of twelve artificial intelligence models, exploring algorithms that seek both linear and non-linear relationships. The Gradient Boosting algorithm demonstrated the best performance in estimating the Permanent Wilting Point, achieving a coefficient of determination (R^2) of 0.74 and a Root Mean Square Error (RMSE) of 0.04 cm^3/cm^3. The project emphasizes the intention to empower experts during the development of the functions, highlighting the relevance of active participation of these professionals throughout all stages of the process.*
Keywords: CRISP-DM, Hybras, Pedotransfer function, Soil Moisture e Data Mine

1. Introdução

Com o avanço tecnológico, na última década a área de Inteligência Artificial (IA) experimentou um notável crescimento, impulsionado pela contínua melhoria na capacidade de processamento das

máquinas. Esta combinação tem permitido a criação de ferramentas e sistemas inovadores para monitorar, prever e mitigar os impactos ambientais. Segundo (CORTéS et al., 2000) a IA pode ser usada de várias maneiras para melhorar a tomada de decisões em diversas áreas de estudos. Uma das maneiras eficientes é por meio do uso de técnicas de interpretação e mineração de dados, que envolvem a triagem de dados para detectar padrões, identificar problemas ou oportunidades potenciais e até mesmo engendrar modelos complexos que contribuem em simulações do mundo real.

As tecnologias de IA são diferentes dos métodos tradicionais de modelagem, pois permitem a omissão de fórmulas matemáticas complexas e informações detalhadas sobre o sistema, sem perda de precisão (YE et al., 2020). Esta facilidade permite a comunidade cientifica adotar a sua utilização, e portanto, trazer grandes benefícios. Segundo (CORTéS et al., 2000), a eficiência e a precisão da análise de dados são melhoradas consideradamente, permitindo tomar decisões embasadas. Além disso, a IA tem a capacidade de processar grandes volumes de dados de forma rápida, identificando padrões que podem passar despercebidos pelos seres humanos. Isso possibilita prever impactos ambientais e encontrar estratégias para mitigá-los.

Contudo, é importante estar ciente das possíveis desvantagens da utilização da IA na tomada de decisões no âmbito ambiental. Sua utilização depende da qualidade e disponibilidade dos dados, e sobretudo da sua abordagem. Além disso, há o risco de viés nos dados ou algoritmos utilizados, o que pode levar a decisões imprecisas ou injustas. A interpretação e explicação dos resultados da análise de IA para pessoas sem conhecimento técnico podem ser desafiadoras.

Neste sentido, existem abordagens sistêmicas que auxiliam o processo de engendramento de modelos, como o CRISP-DM (Cross-Industry Standard Process for Data Mining). A utilização do CRISP-DM permite uma estrutura padronizada, que oferece uma orientação clara nas etapas do desenvolvimento de modelos. Sua abordagem sistemática e iterativa permite adaptações ao longo do processo, tornando os resultados mais robustos. Além disso, o foco no problema de negócios facilita a compreensão, promovendo a colaboração entre diversas partes interessadas, incluindo especialistas de domínio. A ênfase na documentação e transparência é valiosa para a comunicação eficaz, especialmente para aqueles sem conhecimento técnico profundo.

Na agricultura, a determinação da água disponível no solo é crucial em diversas áreas, incluindo a produtividade de agricultura e projetos de irrigação, desempenhando um papel fundamental na agropecuária (SILVA et al., 2008). A determinação desse valor está centrada entre dois pontos-chave: a Capacidade de Campo (CC), que representa o o teor máximo de água que um solo pode reter, e o Ponto de Murcha Permanente (PMP), o ponto crítico em que as plantas não conseguem mais extrair água do solo (OLIVEIRA et al., 2002).

A obtenção desses valores dependem de laboratórios, sendo que a mensuração, especialmente quando envolve a aplicação de pressão, pode ser um processo dispendioso (TOMASELLA; HODNETT; ROSSATO, 2000; HODNETT; TOMASELLA, 2002). Estas técnicas consistem em aplicar pressão ao perfil do solo até que seja drenado toda a umidade. Essa pressão é dada em diferentes potenciais, e o produto dessa mensuração é a criação da curva de retenção do solo (MORAES; LIBARDI; NETO, 1993). Portanto, a cada nível de pressão aplicado, mede-se a quantidade de água que permanece no solo, conhecida como umidade residual. Este processo é repetido em diferentes níveis de pressão. Os autores (FILHO.; CAETANO.; OTTONI., 2022), demonstram que na maioria dos solos o o pico da CC se da em valores potenciais de pressão de 10kpa, enquanto o valor corresponde ao PMP de 1500kpa. Essa curva fornece informações valiosas sobre a capacidade do solo em reter água em diferentes condições, auxiliando na tomada de decisões relacionadas ao manejo da irrigação

e otimização do crescimento das plantas (BARROS et al., 2013).

Conforme destacado por (SAXTON et al., 1986), a relação intrínseca entre as texturas do solo e o conteúdo de água disponível sugere que métodos simples podem ser empregados para estimar essa informação. Essa abordagem ressoa com as discussões anteriores de (BOUYOUCOS, 1951) sobre as propriedades físicas do solo, particularmente em relação à sua granulometria, a qual desempenha um papel crucial na capacidade do solo de reter água.

A introdução do termo Funções de Pedotransferência (PTF) por (BOUMA, 1989) conecta esses conceitos, oferecendo uma perspectiva integrada sobre como simplificar a estimativa de propriedades do solo com base em características cuja obtenção é mais acessível. Há diversos trabalhos sobre o tema, cujo as variáveis independentes são diversas como colocado por (SAXTON et al., 1986).

O autor (OLIVEIRA et al., 2002) descreve como desenvolver fórmulas para calcular características do solo, utilizando como base aquelas que podem ser mais facilmente obtidas como por exemplo as variáveis referentes a granulometria *argila, silte e areia*. Dessa forma, devido à sua sólida fundamentação matemática, surgiu um campo inovador na pedologia. Esse campo visa desenvolver equações que não apenas antecipem o conteúdo de variáveis pedológicas, mas também buscam reduzir custos laborais, acelerar procedimentos analíticos e resolver o problema de não haver muitos laboratórios especializados.

Este trabalho tem como objetivo desenvolver funções de pedotransferência, utilizando o banco de dados HYBRAS, com o propósito de superar os desafios laboratoriais na medição das constantes físico-hídricas. Para atingir esse objetivo de forma eficaz, a aplicação do CRISP-DM se apresentou como uma escolha estratégica. Essa abordagem estruturada e iterativa se contribui no desenvolvimento das PTFs, oferecendo uma metodologia clara e organizada para o desenvolvimento e aprimoramento das funções de pedotransferência.

2. Trabalhos correlatos

Na literatura, a utilização de técnicas de *machine learning* para desenvolver funções de pedotransferência é uma prática ainda pouco utilizada. Isso se deve à complexidade associada à implementação de modelos mais robustos para estimar equações, apresentar os coeficientes de cada atributo e, principalmente, contar com mão de obra especializada. No entanto, com os avanços no campo de inteligência artificial, cada vez mais pesquisadores têm explorado o uso de modelos mais complexos na criação de funções de pedotransferência.

Tabela 1. Trabalhos correlatos.

Autores	Técnicas Escolhidas	Validação cruzada	Hiperparâmetros
(PEREIRA et al., 2018)	*RNA e SVM*	Não	Não
(RAMCHARAN et al., 2017)	*RF*	Sim	Não
(GUNARATHNA et al., 2019)	*RF, RNA e KNN*	Sim	Não
(SEDAGHAT et al., 2022)	*RF, RLM*	Sim	Não
(CATLEY et al., 2009)	Não descreve	Não	Não

Estudos adentram o domínio da ciência de dados, incorporando técnicas mais avançadas de análise. Um exemplo é o trabalho de (PEREIRA et al., 2018), que empregou algoritmos de Redes Neurais Artificiais (RNA) e Máquinas de Vetores de Suporte (SVM) para estimar a resistência do solo usando variáveis pedológicas. Outro estudo relevante foi conduzido por (RAMCHARAN et al.,

2017), que buscou desenvolver equações para estimar a densidade do solo por meio da técnica de Random Forest[1] (RF). As características edáficas foram também exploradas no trabalho de (GUNARATHNA et al., 2019), que estimou o conteúdo de água disponível, incluindo a Capacidade de Campo e o Ponto de Murcha Permanente, utilizando técnicas como RF, Redes Neurais Artificiais e o método dos k-vizinhos mais próximos. O trabalho de (SEDAGHAT et al., 2022) destaca-se ao combinar informações pedológicas, como granulometria e densidade do solo, com índices espectrais de imagens de satélite, utilizando técnicas de RF e Regressão Múltipla. Somente os autores (RAMCHARAN et al., 2017) (GUNARATHNA et al., 2019) e (SEDAGHAT et al., 2022) utilizaram validação cruzada, embora não tenham mencionado a aplicabilidade de hiperparâmetros[3] do modelo, e também empregaram softwares para auxiliar na modelagem. No contexto dos estudos que propõem uma abordagem de estudo de caso, evidenciando a metodologia CRISP-DM, destaca-se a abordagem de (CATLEY et al., 2009), que, embora fuja do escopo da pedologia, é inovadora ao adaptar a abordagem sistemática à sua especificidade. O estudo utiliza dados de natureza fisiológica, coletados em uma Unidade de Terapia Intensiva Neonatal (UTIN) a partir de monitores médicos e dispositivos de suporte à vida, promovendo uma nova metodologia denominada CRISP-TDM.

Portanto, apesar dos trabalhos na área de pedologia apresentarem abordagens metodológicas relevantes para construção dos modelos. Dentre os trabalhos analisados nenhum empregam elementos facilitadores para especialistas que desejam adentrar na área de ciência de dados. Evidenciando a relevância da presente pesquisa, pois, busca demonstrar um estudo de caso empregando técnicas de *machine learning* para criação de funções de pedotransferência, utilizando a metodologia CRISP-DM.

3. Metodologia

Para conduzir a presente pesquisa, foi adotada a metodologia CRISP-DM (Cross-Industry Standard Process for Data Mining). A relevância e a aceitação contínua do CRISP-DM na ciência de dados foram verificadas através do trabalho de (SCHRöER; KRUSE; GóMEZ, 2021), que realizou uma revisão sistemática da literatura selecionando 24 trabalhos recentes e relevantes. O estudo destacou que, mesmo após duas décadas desde o seu lançamento, o CRISP-DM continua sendo amplamente adotado por pesquisadores e profissionais em diversas áreas.

A metodologia escolhida proporciona uma abordagem estruturada e consistente para o desenvolvimento de projetos de análise de dados. Seu caráter sistemático é destacado pela divisão em seis fases: *Compreensão do Negócio, Entendimento dos Dados, Preparação dos Dados, Modelagem, Avaliação e Implantação* distintas como destacado na Figura 1.

[1] A Random Forest cria uma coleção de árvores de decisão, cada uma treinada em uma subamostra aleatória dos dados de treinamento. As previsões individuais de cada árvore são combinadas para produzir uma previsão mais robusta e reduzir o risco de *overfitting*[2], tornando o modelo mais capaz de generalizar para dados não vistos.

[3] Os hiperparâmetros são parâmetros externos ao modelo que influenciam seu comportamento durante o treinamento. Ao contrário dos parâmetros internos, que são aprendidos a partir dos dados, os hiperparâmetros são configurados antes do treinamento e desempenham um papel crucial na modelagem do desempenho do algoritmo.

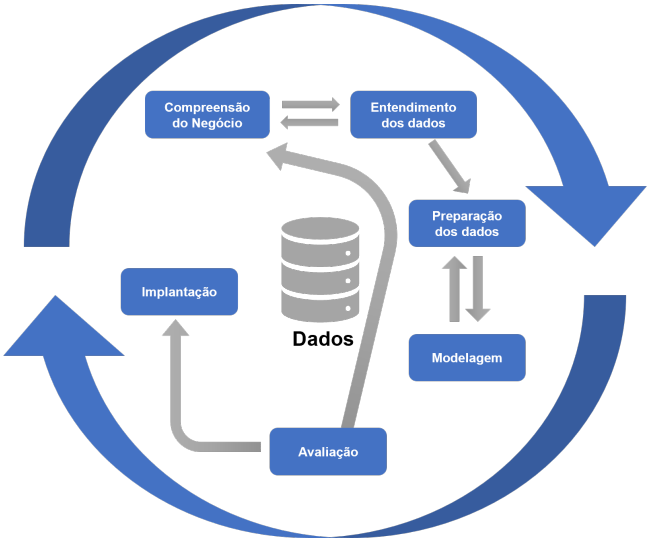

Figura 1. Etapas do CRISP-DM

A escolha do CRISP-DM para guiar o desenvolvimento desta pesquisa se justifica pelo seu caráter bem definido, facilitando a execução sistemática do projeto, bem como pela sua natureza amplamente adotada em diferentes setores e sua independência da indústria específica. Além disso, o CRISP-DM enfatiza a interação entre as etapas promovendo uma relação retroalimentar. Na (figura 2) detalha ainda mais as etapas que foram elaboradas na pesquisa.

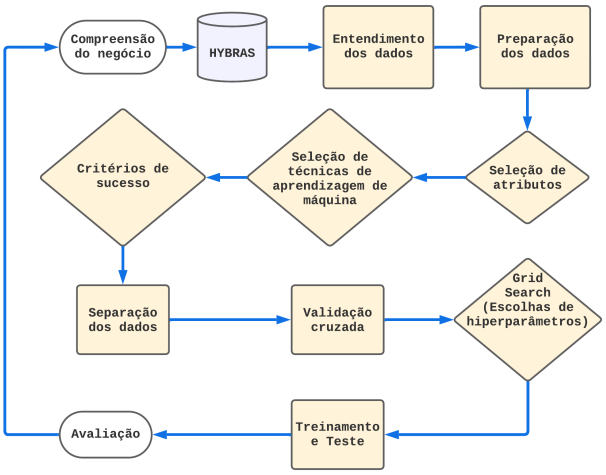

Figura 2. Fluxograma da metodologia da pesquisa

3.1. Banco de dados HYBRAS

Os dados foram extraídos do banco de dados do projeto HYBRAS, que teve como objetivo central consolidar informações relacionadas ao teor de água e condutividade hidráulica saturada (Ks) do solo. Essas informações estão associadas aos atributos fundamentais do solo, bem como aos métodos utilizados para a determinação dessas propriedades (OTTONI et al., 2018). A fundamentação desta proposta é embasada em trabalhos prévios, destacando autores como (TOMASELLA; HODNETT; ROSSATO, 2000; HODNETT; TOMASELLA, 2002). Eles enfatizam as complexidades envolvidas

na aplicação de equações de pedotransferência, especialmente quando essas equações são desenvolvidas com base em dados de solos de regiões tropicais. O desafio reside na variabilidade desses solos em termos de composição e características, o que pode comprometer a precisão e a generalização das equações de pedotransferência para solos de climas tropicais.

As informações foram obtidas de duas tabelas relacionais do HYBRAS, o *SOIL_PROPS e HYDRAULIC_PROPS* que são respectivamente teor de umidade do solo e condutividade hidráulica. Os dados estão distribuídos ao longo de 445 perfis de solos em todo 15 estados brasileiros e contanto com 11 grupos de solos (OTTONI et al., 2018), sendo apenas 102 destes perfis georreferenciados, sendo que cada perfil pode variar de 1 a 12 horizontes. Sendo assim contendo um total de 1075 linhas e 19 colunas. O solo mais representativo do banco de dados é o Latossolo e o Argissolo, que segundo (SANTOS et al., 2018) representam 58% de todo território nacional.

3.1.1. Compreensão e entendimento dos Dados

Dada a natureza contínua dos dados, não foi necessário empregar técnicas para tratar variáveis categóricas. Além disso, destaca-se que os valores não apresentam disparidades significativas que justifiquem a aplicação de técnicas de normalização de dados.

Tabela 2. Descrição estatística para cada variável do solo.

Variável	Média	Mediana	Desvio Padrão	Máximo	Mínimo	Faltantes (%)
code	538.00	538.00	310.47	1075.00	1.00	0.00
clay (%)	36.53	36.00	20.17	96.00	0.00	0.00
silt (%)	21.91	18.80	13.82	63.64	0.00	0.00
sand (%)	41.56	43.00	21.59	97.99	0.40	0.00
vf_sand (%)	6.86	6.40	3.13	16.44	0.60	0.83
f_sand (%)	20.10	16.55	17.08	95.89	0.51	0.58
m_sand (%)	16.28	13.50	10.69	48.32	0.12	0.82
c_sand (%)	14.81	9.05	16.01	67.00	0.00	0.58
vc_sand (%)	2.24	1.10	3.22	15.34	0.00	0.83
ksat (cm/d)	216.83	100.00	387.56	3890.00	0.00	0.60
satwat (cm^3/cm^3)	0.49	0.49	0.10	0.87	0.23	0.27
bulk_den (g/cm^3)	1.39	1.44	0.27	2.01	0.26	0.00
particle_den (g/cm^3)	2.59	2.59	0.18	3.67	1.67	0.08
porosity (cm^3/cm^3)	0.46	0.44	0.11	0.87	0.24	0.08
org_carb (%)	1.51	1.12	1.74	23.40	0.10	0.36
org_mat (%)	2.28	1.80	2.94	40.30	0.08	0.30
102	0.32	0.32	0.09	0.72	0.03	0.03
15300	0.22	0.22	0.09	0.53	0.01	0.01

Observou a presença de valores ausentes, especialmente nas colunas 'vc_sand' e 'vf_sand'. Embora haja técnicas de imputação de dados que empregam cálculos estatísticos simples para preenchimento dessas lacunas, a decisão foi tomar uma abordagem mais conservadora neste estudo exploratório. Optou-se por excluir as colunas com mais de 10% de dados faltantes. Especialmente em contextos pedológicos, onde a finalidade é desenvolver uma equação de pedotransferência para

um comportamento generalista e preciso. De fato, a prática de preenchimento de falhas não é recomendada. Para as outras colunas foram excluídas somente as linhas faltantes.

3.1.2. Preparação dos dados

Nesta etapa, foram escolhidas seis colunas devido sua relação com as variáveis dependentes e por não apresentarem dados faltantes, excluindo deliberadamente a coluna 'code', utilizada unicamente para a identificação dos perfis no banco de dados. Ao explorar a (figura 3), não foram identificados valores atípicos, o que está em concordância com os intervalos definidos por (SANTOS et al., 2018). A coluna 'particle_den' apresentou amostras consideradas *outliers* pelo método do intervalo interquartil, contudo foram pouquíssimos dados ao observar o histograma da (figura 4). Entretanto, é crucial observar que esses valores atípicos demonstram uma forte correlação com outras duas variáveis, 'bulk_den' e 'porosity'. Apesar que essa relação fica ainda mais significativa quando comparado entre as duas, como visto na matriz de correlação apresentada na (figura 3.

Tabela 3. Descrição das variáveis.

	Descrição
clay	Argila (%)
silt	Silte (%)
sand	Areia (%)
bulk_den	Densidade do solo (g/cm³)
particle_den	Densidade da particula (g/cm³)
porosity	Porosidade (%)
102	Teor de água CC (cm³/cm³)
15300	Teor de água PMP (cm³/cm³

De acordo com (HILLEL, 2003) essa correlação, é a influência do método de medição da porosidade, cuja relação matemática é expressa por:

$$Porosidade = 100 - \left(\frac{Densidade\ do\ Solo}{Densidade\ de\ Partícula} \right) \times 100 \tag{1}$$

Conforme delineado por (DONAGEMMA et al., 2011), a densidade de partícula é definida como a relação entre a massa de uma amostra de solo e o volume que ela ocupa, desconsiderando a porosidade. Essa densidade mostra pouca variação e geralmente permanece em torno de 2,65 g/cm³. Essa estabilidade é atribuída, em grande parte, à consistência dos minerais presentes no solo, os quais não tendem a variar significativamente.

A granulometria (figura 4) de um solo refere-se à distribuição das partículas de areia, silte e argila presentes. É relevante observar que a granulometria é expressa em porcentagem, implicando que a soma dessas classes deve totalizar 100% (SANTOS et al., 2018). Assim, as informações obtidas estão em concordância com a realidade do conjunto de dados.

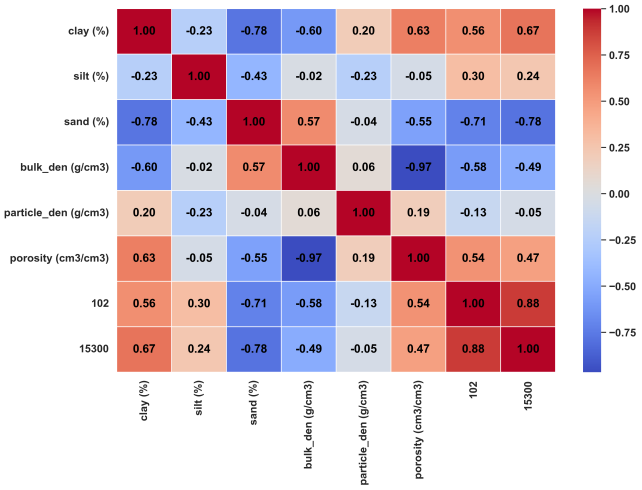

Figura 3. Matriz de correlação.

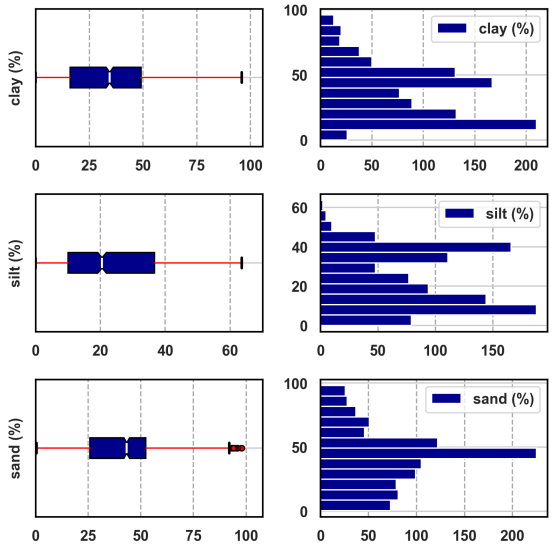

Figura 4. Distribuição da granulometria.

Vale ressaltar a importância de analisar a distribuição das faixas granulométricas, especialmente no que diz respeito ao silte. Conforme destacado por (TOMASELLA; HODNETT; ROSSATO, 2000), em regiões com climas mais temperados, os solos têm maior presença de partículas de silte. Essa disparidade na distribuição granulométrica pode influenciar nas predições do modelo e deve ser cuidadosamente considerada na interpretação dos resultados, pois os solos de clima tropical contém mais partículas arenosas devido sua mineralogia diferente dos climas temperados.

A matriz de correlação (figura 3) demonstra as relações lineares entre as variáveis independentes e as variáveis dependentes. Indicando que resultados próximos de zero não há uma relação linear forte. No entanto, é importante lembrar que a correlação não captura todas as formas de relação entre as variáveis. Pode haver outras relações não lineares ou complexas que não são refletidas pela correlação. Em consideração as variáveis dependentes '102Kpa' (CC) e '15300Kpa' (PMP) podemos observar uma relação forte entre valores de granulometria, densidade e porosidade.

A presença de multicolinearidade, como previamente evidenciado entre densidade e porosi-

dade, pode acarretar uma série de efeitos prejudiciais para as regressões lineares, conforme descrito por (COIMBRA et al., 2005). Isso inclui a possibilidade de estimativas incorretas dos coeficientes de regressão, bem como a superestimação dos efeitos diretos das variáveis independentes sobre a variável dependente. A multicolinearidade, um fenômeno estatístico que se manifesta quando duas ou mais variáveis independentes em um modelo de regressão apresentam alta correlação, revela uma relação linear significativa entre essas variáveis.

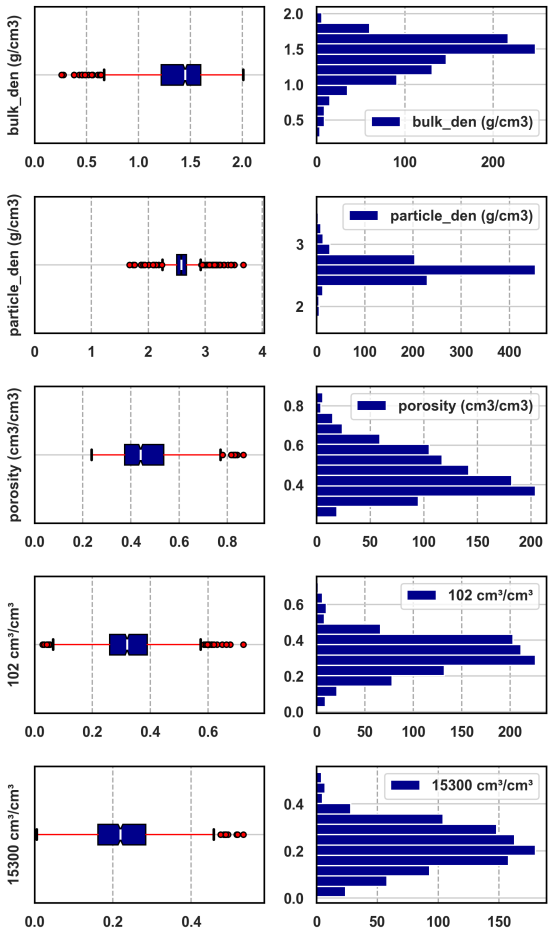

Figura 5. Distribuição das variáveis independentes.

Os impactos da multicolinearidade se estendem por diversas áreas nas análises de regressão e modelagem estatística, gerando implicações negativas. A instabilidade nas estimativas dos coeficientes surge como um desafio, pois a presença desse fenômeno pode tornar os coeficientes de variáveis correlacionadas instáveis e de difícil interpretação. Variações mínimas nos dados podem resultar em oscilações acentuadas nas estimativas (YOO et al., 2014).

Além disso, a multicolinearidade dificulta a identificação do impacto individual de cada variável, uma vez que parte da variação em uma variável pode ser explicada pela correlação com outra. As variâncias dos coeficientes de variáveis correlacionadas aumentam, indicando menor precisão nas estimativas. Essa condição afeta a inferência estatística, comprometendo a validade de intervalos de confiança e testes de significância (SHRESTHA, 2020).

3.2. Modelagem

Os objetivos do presente estudo se relacionam com a etapa de compreensão da regra do negócio da metodologia CRISP-DM, que por sua vez busca discutir e avaliar as técnicas de *machine learning* escolhidas no engendramento de funções de pedotransferência. A escolha das técnicas de regressão foi baseada na capacidade de capturar relações lineares e não lineares, lidar com multicolinearidade e *overfitting*, interpretar coeficientes de regressão, tratar dados ausentes e melhorar a precisão geral do modelo.

Antes de iniciar a modelagem, foram feitas as escolhas dos hiperparâmetros para cada modelo, empregando a técnica de *Grid Search*[4]. Está técnica foi utilizado para explorar exaustivamente todas as combinações paramétricas do modelo. A métrica escolhida para guiar esse processo foi o *neg_mean_squared_error*, indicando ao *Grid Search* que deveria validar modelos nos quais o erro médio quadrático é maximizado negativamente, ou seja, minimizando o erro quadrático médio (PEDREGOSA et al., 2011). A seleção dos hiperparâmetros, conforme mencionado, foi realizada por meio de validação cruzada, assegurando uma escolha robusta e generalizável.

3.2.1. Separação dos dados

Para realizar essa análise, o conjunto de dados foi estratificado em subconjuntos de treinamento e teste, seguindo uma proporção de 80% e 20%, respectivamente. Essa divisão não apenas permitiu a avaliação das variáveis principais CC e PMP, respectivamente '102Kpa' e '15300Kpa', mas também possibilitou a identificação de padrões e relações não lineares entre essas variáveis.

A escolha de dividir os dados em 80% para treinamento e 20% para avaliação cruzada foi motivada não apenas pela prática comum, mas também respaldada pelo estado da arte em funções de pedotransferência. Autores como (TOMASELLA; HODNETT; ROSSATO, 2000), (HODNETT; TOMASELLA, 2002), (OLIVEIRA et al., 2002), (SILVA et al., 2008), (BARROS et al., 2013), utilizaram essa abordagem de particionamento em seus estudos, destacando sua eficácia na modelagem de fenômenos complexos. Essa estratégia não apenas mantém a consistência com as práticas estabelecidas, mas também proporciona uma base sólida para a validação e generalização do modelo.

3.2.2. Validação cruzada

A validação cruzada é uma técnica crucial no desenvolvimento de modelos de aprendizado de máquina. Ela desempenha um papel fundamental na avaliação do desempenho e na seleção dos melhores hiperparâmetros do modelo. Segundo (BERRAR, 2018) a validação cruzada em vez de dividir o conjunto de dados em apenas dois conjuntos, treinamento e teste, a validação cruzada divide o conjunto de dados em vários subconjuntos. O modelo é treinado em alguns subconjuntos e avaliado em outros. Esse processo é repetido várias vezes, garantindo que o modelo seja testado em diferentes subconjuntos dos dados.

O mesmo autor avalia a confiabilidade do emprego dessa técnica, mostrando-se imprescindível no desempenho do modelo, reduzindo a probabilidade de superestimar ou subestimar a precisão do mesmo. Com base nos resultados da validação cruzada, é possível obter uma estimativa mais realista de como o modelo irá se comportar em dados desconhecidos. Portanto, esta técnica evidencia

[4] O *Grid Search* é uma abordagem exaustiva que explora todas as combinações possíveis de valores de hiperparâmetros especificados pelo usuário.

relações não lineares que podem desempenhar um papel crucial na compreensão mais aprofundada do comportamento dos dados. Essas relações não lineares são de importância significativa para obter análises mais precisas durante a fase de modelagem, contribuindo para uma representação mais fiel da complexidade subjacente nos dados.

Muitos algoritmos de aprendizado de máquina têm hiperparâmetros que precisam ser configurados para que o modelo seja mais eficaz. Uma abordagem fácil e bastante adotada dependente do tamanho da base de dados é o *Grid Search*, de acordo com (LAVALLE; BRANICKY; LINDEMANN, 2004) que definem a aplicação do *Grid Search*, onde é criada uma grade com todas as combinações possíveis dos valores dos hiperparâmetros que se deseja otimizar. O modelo é treinado e avaliado usando validação cruzada para cada combinação de hiperparâmetros na grade. Ao final, é obtido o conjunto de hiperparâmetros como a melhor configuração do o modelo. O emprego desta técnica nos trabalhos supracitados não foi identificado.

3.2.3. Técnicas de machine learning escolhidas

Houve a escolha de 4 técnicas de aprendizagem de máquina, capaz de lidar com relação lineares e não lineares. Modelos baseados em árvores de decisão, conforme discutido por (PODGORELEC et al., 2002), destacam-se ao lidar com dados caracterizados por forte correlação entre variáveis independentes, uma vez que não pressupõem uma relação linear entre essas variáveis e a variável dependente. O RF capitaliza essa característica, atuando como uma coleção de árvores de decisão. Cada árvore é treinada em uma amostra aleatória dos dados, introduzindo variabilidade entre elas. Durante a construção, apenas um subconjunto aleatório de características é considerado, fomentando diversidade e evitando *overfitting*.

As previsões de cada árvore são combinadas por meio de voto majoritário para problemas de classificação ou por média para problemas de regressão. A RF destaca-se pela robustez, desempenho consistente em grandes conjuntos de dados e facilidade de aplicação, sem a necessidade de extenso ajuste de parâmetros. Apesar de cada árvore ser menos interpretável individualmente, a análise conjunta das árvores permite investigações abrangentes sobre as relações entre as características e a variável de destino, conforme proposto por (BREIMAN, 2001; SAHOUR et al., 2021).

Gradient Boosting, conforme estudado por (SAHOUR et al., 2021), é outra técnica relevante. Essa abordagem, similar ao RF em muitos aspectos, aprimora o desempenho do modelo ao ajustar iterativamente as árvores, focando nos erros residuais do modelo anterior. Isso confere ao Gradient Boosting uma capacidade robusta de lidar com dados complexos, superando limitações e melhorando a precisão em relação ao RF, especialmente em situações onde a melhoria incremental é essencial.

Equação geral do Gradient Boosting:

$$F(x) = F_0(x) + \eta \cdot h_1(x) + \eta \cdot h_2(x) + \ldots + \eta \cdot h_T(x) \tag{2}$$

onde:

$F(x)$ é a função de predição final,
$F_0(x)$ é a predição inicial,
b é a taxa de aprendizado que controla a contribuição de cada modelo,
$h_(x)$ são árvores de decisão individuais.

O SVR por se tratar de um algoritmo da família de Support Vector Machines (SVM) trabalha com margens de separação, e portanto altamente recomendados para relações complexas de não linearidades, sua aplicação foi realizada em um estudo feito por (XIA et al., 2017) onde discorre sobre a escolha do SVR devido sua capacidade de lidar com dados ruidosos e é menos suscetível a *overfitting* do que outros algoritmos de regressão.

Equação da SVR:

$$f(x) = \sum_{i=1}^{N} \alpha_i \cdot K(x, x_i) + b \tag{3}$$

onde:

f(x) é a função de predição para a entrada x,
N é o número de vetores de suporte,
α_i são os coeficientes que são determinados durante o treinamento,
$K(x, x_i)$ é a função de kernel, que calcula a similaridade entre os vetores de entrada,
b é o termo de viés.

A Regressão Múltipla é uma técnica estatística que estabelece uma relação linear entre uma variável dependente e várias variáveis independentes. É frequentemente utilizada para análises estatísticas e previsões. Além disso, a regressão linear é o alicerce para muitas ferramentas de modelagem modernas (SU; YAN; TSAI, 2012).

Equação da Regressão Múltipla (com uma variável independente):

$$RM = \beta_0 + \beta_1 \cdot X_1 + \beta_2 \cdot X_2 + ... + \beta_n \cdot X_n + \varepsilon \tag{4}$$

onde:

Y é a variável dependente,
X é a variável independente,
n é o número das variáveis independentes,
β_0 é o intercepto da linha de regressão,
ε é o termo de erro, que captura a variação.

Trabalhos anteriores, como os de (TOMASELLA; HODNETT; ROSSATO, 2000), (HODNETT; TOMASELLA, 2002), (SILVA et al., 2008), e (BARROS et al., 2013), utilizaram a regressão linear como um dos modelos para determinar seus coeficientes. No entanto o artigo de (SHRESTHA, 2020) demonstra que os modelos de regressões lineares Ridge e Lasso adicionam uma penalidade à função de perda do modelo de regressão, reduzindo assim a influência das variáveis altamente correlacionadas. Isso ajuda a estabilizar os coeficientes de regressão e reduzir a sensibilidade aos dados.

3.2.4. Seleção de características

Neste estudo, as técnicas de aprendizagem de máquina supracitadas foram obtidos por meio da biblioteca *sklearn*. Entre as quatro técnicas aplicadas, utilizamos seis algoritmos distintos. Como mencionado anteriormente, identificamos fortes relações entre as variáveis independentes. A presença

de multicolinearidade, destacada nesse contexto, pode impactar significativamente o desempenho de técnicas de regressão que pressupõem relações lineares. A multicolinearidade surge quando as variáveis independentes do modelo estão altamente correlacionadas, o que pode complicar a interpretação dos coeficientes e prejudicar a estabilidade das estimativas. Esse fenômeno merece atenção especial durante a análise e modelagem, pois pode afetar a qualidade e a confiabilidade dos resultados obtidos.

Paral tal, foi utilizando técnicas de seleção de característica como a matriz de correlação, utilização de algoritmos que penalizam características de baixa relação como regressão linear Lasso e Ridge (SHRESTHA, 2020) e a técnica de Eliminação Recursiva de Atributos (RFE), tendo o objetivo de selecionar características considerando conjuntos cada vez menores de variáveis, com base em um estimador externo que atribui pesos às características, como os coeficientes de um modelo linear. Esse procedimento é repetido recursivamente no conjunto podado até que o número desejado de características seja atingido (PEDREGOSA et al., 2011).

No contexto da Eliminação Recursiva de Atributos (RFE), identificou-se que a característica de maior peso foi a presença de areia em relação ao conteúdo de água disponível. Essa associação é notavelmente destacada na matriz de correlação (figura 3). Contudo, é interessante observar que ao utilizar o algoritmo Ridge, os coeficientes indicam que a porosidade emerge como a característica mais influente. Essa análise ilustra a complexidade das relações entre as características do solo, evidenciando como diferentes métodos podem apontar para variáveis distintas como sendo mais relevantes em um determinado contexto.

Assim, adotamos uma abordagem iterativa, comumente conhecida como *backward*[5], começando o treinamento com todas as variáveis independentes e eliminando uma a cada iteração (SUTTER; KALIVAS, 1993). Essa análise possibilitou a compreensão das características inerentes de cada algoritmo, especialmente no que diz respeito às relações lineares e não lineares entre as variáveis. Ao conduzir esse processo iterativo, foi possível examinar como cada variável contribui para o modelo, destacando nuances importantes na natureza das relações exploradas pelos algoritmos. Portanto, foi selecionado 3 cenários que demonstram a complexidade das variáveis independentes, o cenário com todas as variáveis, outro com apenas apenas 2 *sand e bulk_den* e a ultima com apenas uma *sand*.

3.3. Critérios de sucesso

Foram selecionados alguns trabalhos de relevância dentro o contexto de funções de pedotransferência (TOMASELLA; HODNETT; ROSSATO, 2000; OLIVEIRA et al., 2002; BARROS et al., 2013) para estabelecimento de critérios de sucesso semelhantes, onde modelos de regressão linear são frequentemente utilizados para representar os coeficientes das variáveis independentes.

Nesses trabalhos, observa-se que um coeficiente de determinação R2 superior a 0.7 e um RMSE abaixo de 0.06 cm³/cm³ são considerados indicadores de qualidade das funções de pedotransferência. Esses critérios fornecem um filtro robusto para avaliar a precisão e confiabilidade dos modelos, garantindo que eles sejam capazes de explicar a variação nos dados de forma adequada e sobretudo um comportamento generalista.

Portanto, foram selecionados quatro métricas para avaliação dos modelos:

MAE (Erro Absoluto Médio):

[5] A seleção de características "backward"é uma estratégia iterativa que visa melhorar a eficiência computacional e a generalização do modelo, eliminando características menos informativas ou redundantes.

O Erro Absoluto Médio (MAE) é uma métrica usada para medir a média da diferença absoluta entre os valores reais (Y) e os valores previstos ($\hat{Y}$). Ele nos dá uma ideia de quão próximas as previsões estão dos valores verdadeiros, em média. Quanto menor o MAE, melhor o desempenho do modelo (CHAI; DRAXLER, 2014). O MAE é fácil de interpretar, pois representa o desvio absoluto médio em relação aos valores reais.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|Y_i - \hat{Y}_i| \tag{5}$$

onde:

n é o número total de observações,
Y é o valor real observado,
$\hat{Y}$ é o valor previsto.

RMSE (Raiz do Erro Quadrático Médio):

O Erro Quadrático Médio (RMSE) é outra métrica popular usada para avaliar a precisão das previsões. O RMSE calcula a raiz quadrada da média dos quadrados das diferenças entre os valores reais (Y) e os valores previstos ($\hat{Y}$) (CHAI; DRAXLER, 2014). O RMSE é sensível a erros maiores e penaliza-os mais do que erros menores, tornando-o útil para detectar grandes discrepâncias entre as previsões e os valores verdadeiros.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2} \tag{6}$$

onde:

n é o número total de observações,
Y é o valor real observado,
$\hat{Y}$ é o valor previsto.

MAPE (Erro Percentual Absoluto Médio):

O MAPE é calculado pela média percentual dos erros absolutos entre as previsões e os valores reais. De acordo com (MYTTENAERE et al., 2016) O MAPE é frequentemente utilizado na prática devido à sua interpretação muito intuitiva em termos de erro relativo. O autor (MORENO et al., 2013) observou que quando os valores reais forem próximo de zero, o MAPE pode não ser uma medida confiável de precisão.

A fórmula do MAPE é dada por:

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{Y_i - \hat{Y}_i}{Y_i}\right| \tag{7}$$

onde:

n é o número total de observações,
Y é o valor real observado,
$\hat{Y}$ é o valor previsto.

R² (Coeficiente de Determinação):

O Coeficiente de Determinação, comumente representado por R², é uma métrica que representa a proporção da variância na variável dependente (Y) que é previsível a partir da variável independente ($\hat{Y}$) pelo modelo (HEMPHILL, 2003). Em outras palavras, mede o quão bem o modelo se ajusta aos dados em comparação com uma média simples (média aritmética). O R² varia de 0 a 1, onde 0 indica que o modelo não explica nenhuma variabilidade nos dados e 1 indica um ajuste perfeito. O autor (CHICCO; WARRENS; JURMAN, 2021) observou-se que o coeficiente de determinação quantifica a capacidade do modelo de ajustar-se aos dados observados e capturar a variação na variável de interesse, mais preciso do que o MSE e MAE.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2}{\sum_{i=1}^{n}(Y_i - \bar{Y})^2} \tag{8}$$

onde:

n é o número total de observações,
Y é o valor real observado,
$\hat{Y}$ é o valor previsto pelo modelo,
$\bar{Y}$é a média dos valores reais observados.

Para determinar o sucesso dos modelos, foram considerados aqueles que apresentaram o menor erro (MAE, RMSE, MAPE) e o maior coeficiente de determinação (R2).

4. Resultados e Discussão

A primeira iteração, na qual todas as variáveis foram utilizadas conforme representado nas figuras 6 e 7, revela valores de erros que se mostram dentro da faixa aceitável, conforme estabelecido pela literatura mencionada sobre funções de pedotransferência. Os valores de RMSE, por exemplo, situam-se abaixo de 0.06 cm³/cm³ na estimativa para a variável no potencial mátrico de 1500 cm³/cm³.

4.1. Primeiro cenário

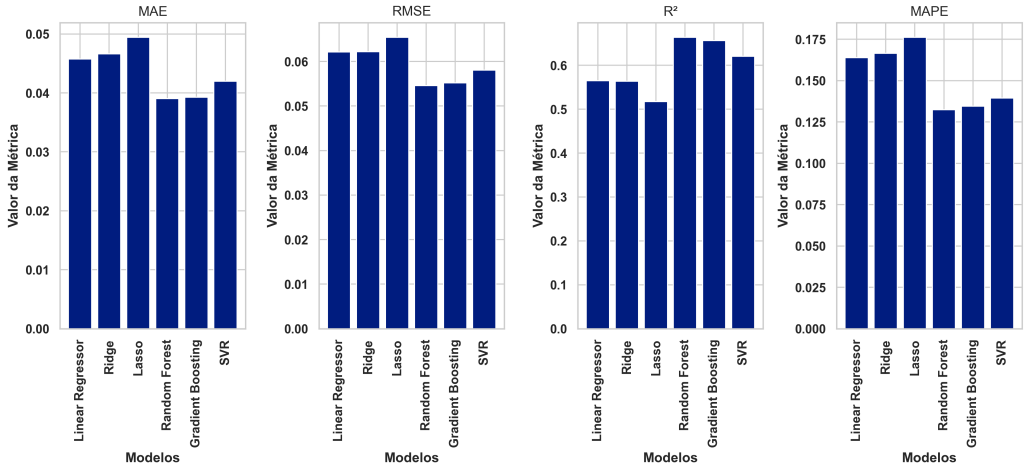

Figura 6. Métricas de capacidade de campo utilizando todas as variáveis independentes

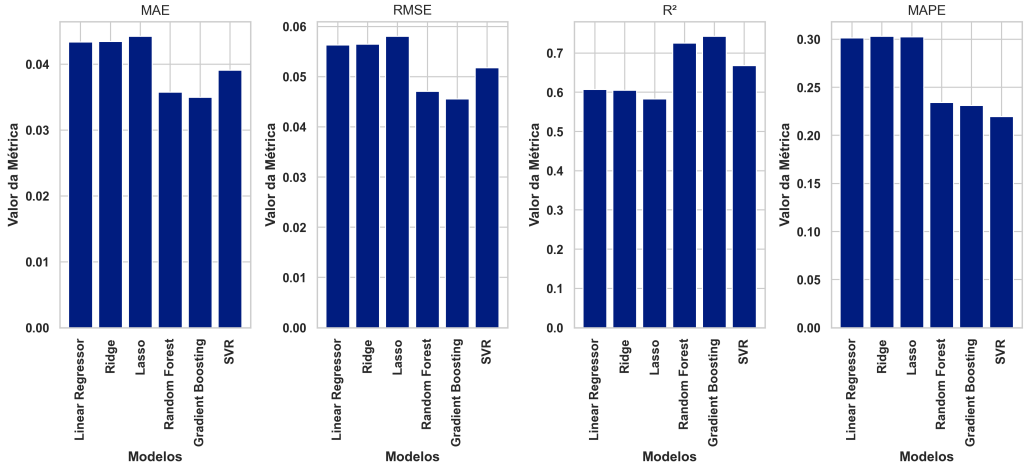

Figura 7. Métricas de ponto de murcha permanente utilizando todas as variáveis independentes.

Entretanto, é importante notar que, nessa configuração, a métrica MAPE apresenta os maiores erros em comparação com a estimativa da variável no potencial mátrico de 102 cm³/cm³. Os algoritmos baseados em relações lineares mostraram-se particularmente desafiadores na previsão, exibindo os maiores erros e o menor coeficiente de determinação.

Dentre esses algoritmos, o Lasso destacou-se negativamente, demonstrando a menor performance. Isso é atribuído à sua natureza de zerar os coeficientes que não possuem uma relação linear forte. Nesse contexto, todas as variáveis, exceto teores de *sand*, foram multiplicadas por 0 na tentativa de favorecer o desempenho do modelo. Contudo, esse ajuste não foi suficiente para melhorar as métricas de avaliação.

É relevante observar que, apesar dessas dificuldades, o MAE permaneceu abaixo de 0.04 cm³/cm³, indicando um comportamento específico do modelo em relação à média absoluta dos erros.

Esse resultado sugere uma tendência do modelo em manter um desempenho consistente mesmo diante das complexidades identificadas na previsão de determinadas variáveis.

Nesse contexto específico, a escolha do melhor modelo varia de acordo com a variável alvo em questão. Ao analisar a estimativa do potencial mátrico de 15300 cm³/cm³, observamos que o modelo SVR se destaca, apresentando o menor MAPE entre as opções consideradas. Já no caso do potencial mátrico de 102 cm³/cm³, o modelo RF exibe o menor MAPE de 13%, destacando-se como a escolha mais precisa para essa condição específica. Notavelmente, o modelo Gradient Boosting demonstra desempenho superior em comparação com outras métricas avaliadas, consolidando-se como a opção mais eficaz para a variável 15300 cm³/cm³ MAPE de 23%, RMSE 0.05 cm³/cm³, MAE 0.04 cm³/cm³ e R^2 de 0.74.

4.2. Segundo cenário

Neste cenário, optou-se por remover colunas que apresentavam multicolinearidade, visando favorecer a relação linear. Além disso, foram excluídas as colunas que demonstravam relações mais fracas com as variáveis alvo, modelando somente os atributos de *sand e bulk_den*. Um aspecto distintivo desse cenário é a busca por um modelo capaz de estimar valores de água disponível usando dados de fácil mensuração e baixo custo laboral, como areia e densidade do solo (SANTOS et al., 2018) (DONA-GEMMA et al., 2011). Apesar dos esforços empregados, os algoritmos de regressão linear ainda apresentam as métricas mais modestas. A comparação com as métricas previamente mencionadas evidencia a deterioração do modelo nesse cenário. Notavelmente, o algoritmo de RF emerge como o mais eficaz, exibindo os menores erros e o maior coeficiente de determinação, consolidando-se como o modelo mais robusto diante das complexidades inerentes ao conjunto de dados.

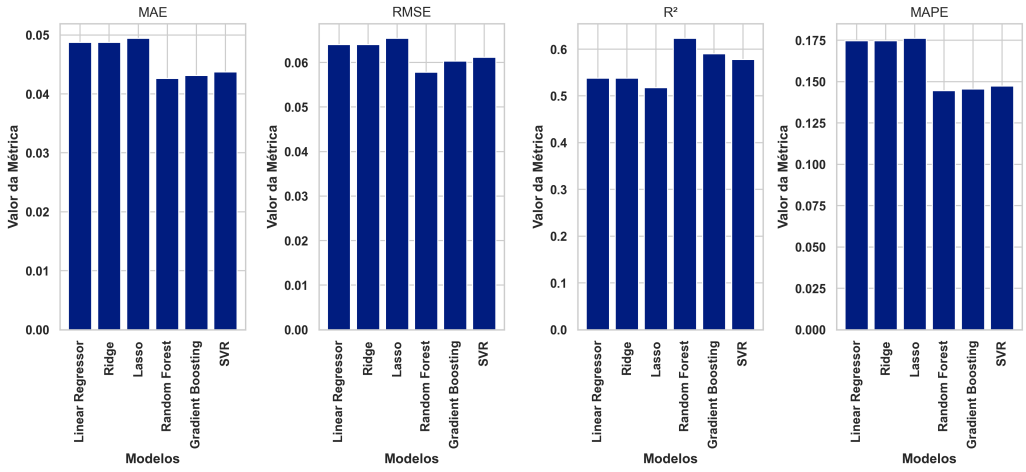

Figura 8. Métricas de capacidade de campo utilizando somente as colunas *sand e bulk_den* como variável independente.

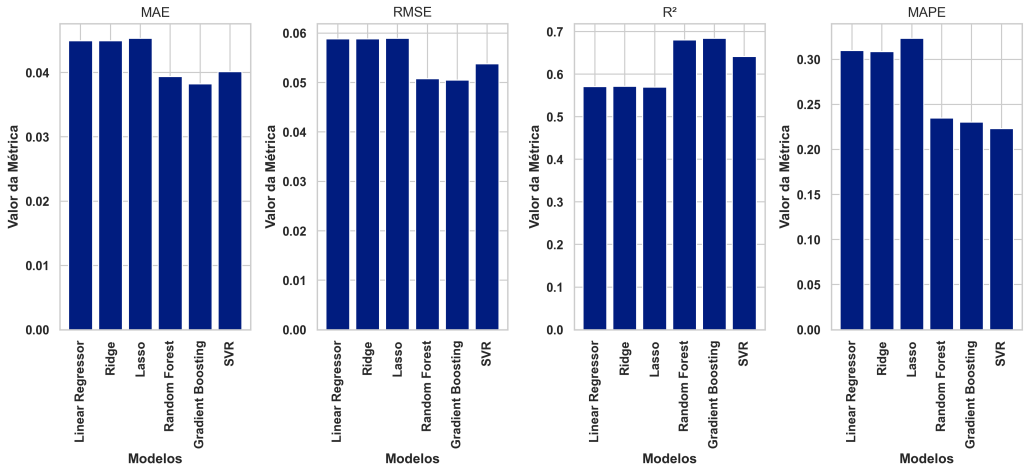

Figura 9. Métricas de ponto de murcha permanente utilizando somente as colunas *sand* e *bulk_den* como variável independente.

4.3. Terceiro cenário

Na Figura 10 e 11 temos a modelagem com somente a coluna *sand*, pode-se observar que os menores erros estão entre as técnicas de modelagem que conseguem captar as relações não lineares, e sobretudo, o maior coeficiente de determinação.

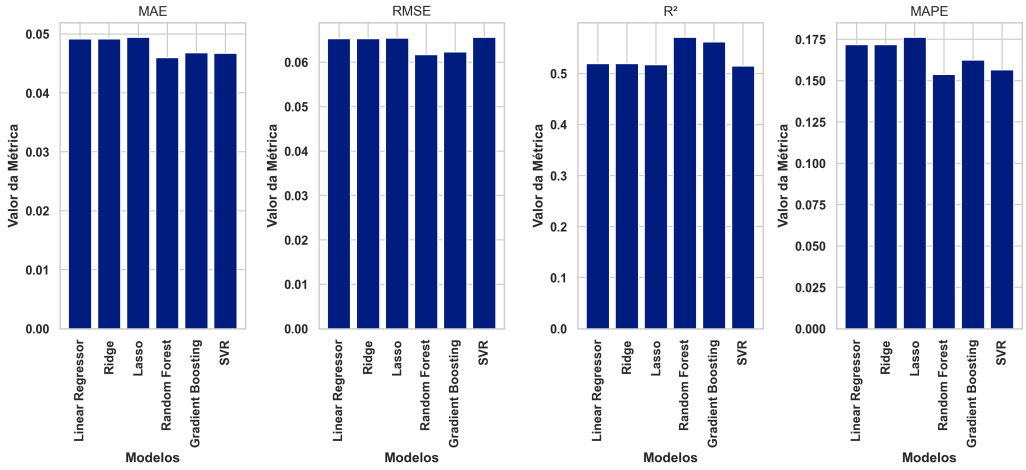

Figura 10. Métricas de capacidade de campo utilizando somente a coluna *sand* como variável independente.

Nenhum algoritmo ficou abaixo do RMSE de 0.06 cm³/cm³, métrica comum entre os trabalhos da literatura. O erro percentual, quando queremos prever o potencial PMP chega próximo de 30%. Apesar de reduzir a dimensionalidade afim de contribuir com técnicas com análise linear, os algoritmos de regressão não apresentaram boas métricas. Validando a ideia que as relações intrínsecas entre a variável alvo transcende relações lineares simples.

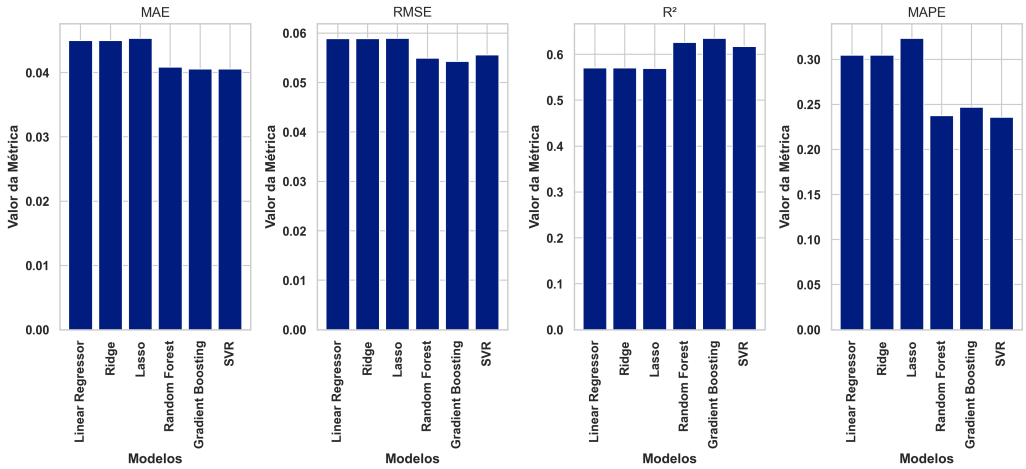

Figura 11. Métricas de ponto de murcha permanente utilizando somente a coluna *sand* como variável independente.

4.4. Avaliação

Em resumo, os resultados deste estudo destacam a eficácia dos modelos baseado em árvores, especialmente o RF, para lidar com as complexidades e não linearidades presentes nas funções de pedotransferência. Os modelos de regressão linear tiveram uma precisão inferior aos demais e obtiveram os maiores erros, no entanto, mesmo com coeficientes de determinação abaixo de 0.7, ainda podem ser úteis, fornecendo métricas de erro aceitáveis e permitindo o cálculo dos coeficientes da equação. O que pode ser útil para calcular os valores alvo da modelagem sem a necessidade de implantação do modelo. Na tabela 4 temos a relação do desempenho do melhor cenário, no qual foi modelado com todas as variáveis.

Tabela 4. Desempenho dos Modelos de Regressão para CC e PMP.

Modelo	RMSE_CC	RMSE_PMP	R^2_CC	R^2_PMP
Linear Regressor	0.062	0.056	0.564	0.607
Ridge	0.062	0.056	0.563	0.605
Lasso	0.065	0.058	0.517	0.582
Random Forest	0.054	0.047	0.663	0.725
Gradient Boosting	0.055	0.045	0.656	0.742
SVR	0.058	0.051	0.620	0.668

Os modelos de ensemble RF e Gradient Boosting se destacaram como os mais adequados de acordo com as métricas obtidas a partir dos artigos de referência, especialmente os estudos de autores na área de funções de pedotransferência. Esses resultados são encorajadores, pois os modelos de árvores são conhecidos por sua capacidade de lidar com relacionamentos complexos e não lineares entre as variáveis independentes e a variável dependente, o que é essencial ao lidar com dados de funções de pedotransferência.

Além disso, o SVR também apresentou resultados interessantes, com métricas próximas aos modelos de árvore de decisão. Essa similaridade pode ser atribuída à presença de colinearidade entre as variáveis independentes e à relação não linear dos dados. Essas características são bem tratadas por algoritmos de árvores e, portanto, também refletem nos resultados do SVR.

O desempenho dos modelos neste estudo mostrou-se próximo do esperado na literatura, considerando as distintas características dos dados, influenciadas pelas especificidades regionais do Brasil. O banco de dados do HYBRAS apresentou uma diversidade significativa, com 445 perfis distribuídos em todo o território nacional, concentrando-se principalmente nas regiões sul e sudeste. Essa diversidade trouxe desafios para alguns modelos, tornando a obtenção de bons resultados mais complexa.

Diante dessa complexidade, foram selecionados cuidadosamente 12 modelos para análise, buscando compreender melhor o comportamento geral da base de dados. O trabalho realizado foi bem-sucedido, resultando na criação de 12 funções de pedotransferência capazes de estimar a água disponível do solo em dois potenciais, 102 (CC) e 15300 (PMP).

Portanto, este trabalho superou as dificuldades enfrentadas, desenvolvendo funções de pedotransferência relevantes para estimar a água disponível do solo.

A diversidade do banco de dados do HYBRAS exigiu escolhas criteriosas dos modelos, e apesar das limitações encontradas, os resultados obtidos são valiosos para a compreensão das relações entre as variáveis do solo e a água disponível em diferentes regiões do Brasil.

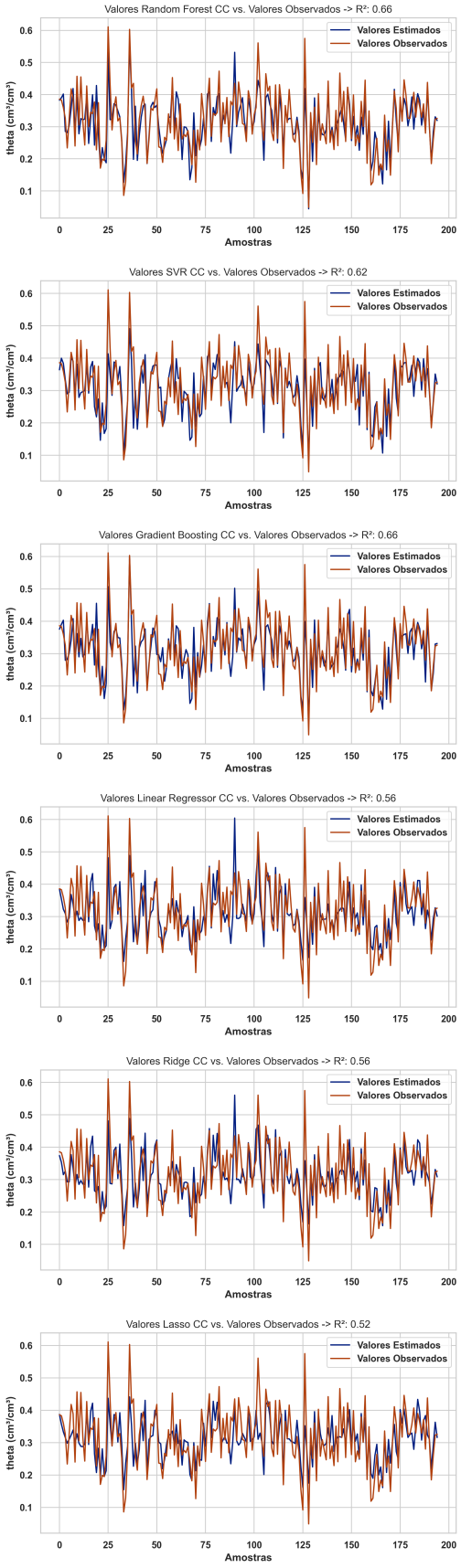

Figura 12. Comparação de estimativas do modelo CC em relação aos dados observados

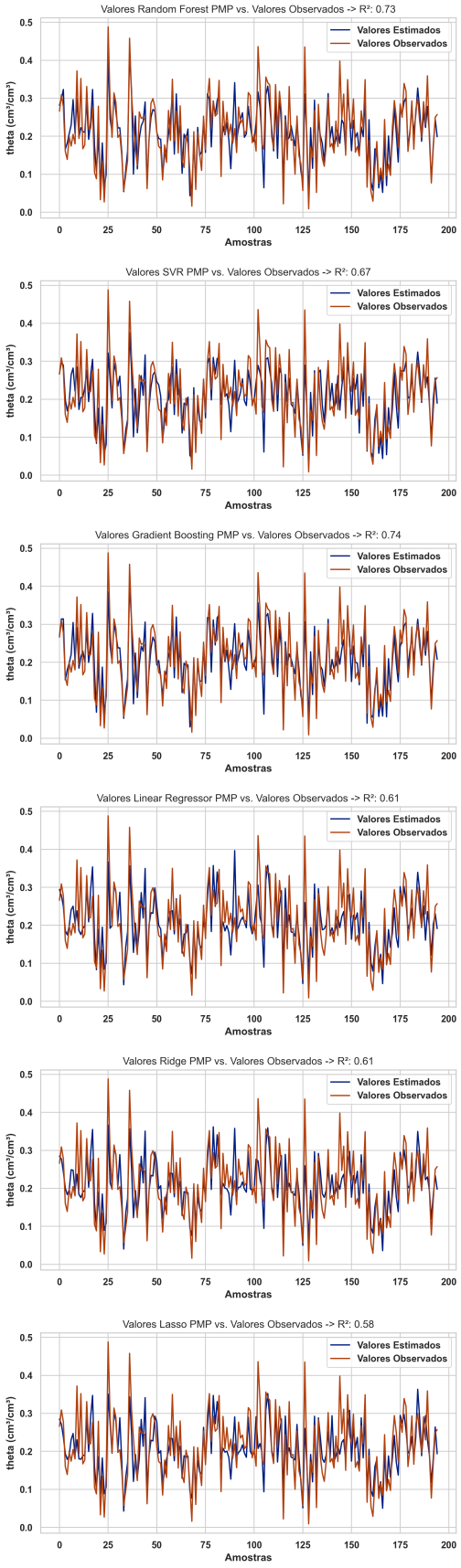

Figura 13. Comparação de estimativas do modelo PMP em relação aos dados observados

5. Considerações Finais

A abordagem metodológica utilizada no CRISP-DM centrada na constante avaliação da situação, buscando continuamente o alinhamento com os objetivos de negócio encoraja a comunicação entre as partes interessadas e o profissional de modelagem de dados. Muito envolta das entregas específicas para cada parte do processo, e elucidações de dúvidas no decorrer da modelagem. Essa dinâmica favorece a precisão do resultado final. Contudo, a comunicação as vezes não é o bastante para solucionar os problemas de modelagem. Diante disto, a metodologia do CRISP-DM capacita os especialistas de negócio a aplicarem suas próprias técnicas de modelagem. Os profissionais podem realizar análises de dados de maneira mais sistemática, garantindo uma compreensão abrangente do problema em questão e uma implementação mais eficiente das soluções propostas.

Os modelos que obtiveram melhores avaliações do primeiro cenário estão prontos para serem utilizados em implementações práticas. No entanto, como trabalho futuro, é recomendado realizar uma seleção mais específica de dados, considerando informações sobre a ordem textural ou tipo de solo. Essa abordagem contribuiria significativamente para aperfeiçoar o desempenho do modelo, seguindo exemplos de estudos anteriores, como o trabalho realizado por (OLIVEIRA et al., 2002), que utilizou dados de solos do estado de Pernambuco para criar funções de pedotransferência em diferentes hierarquias.

A abordagem de refinamento utilizando dados específicos permitirá que o modelo seja mais adequado e preciso em relação a diferentes características texturais ou tipos de solo encontrados em diferentes regiões do Brasil. Ao considerar essas nuances, o modelo poderá ser ajustado de forma mais precisa para atender às particularidades de cada contexto.

Embora alguns modelos tenham apresentado métricas satisfatórias, é importante destacar que o processo de melhoria contínua e aprimoramento dos modelos são essenciais para alcançar resultados ainda mais precisos e confiáveis. Deste, modo analisar o tempo de treinamento do modelo como uma métrica de escolha é altamente recomendável. Ao realizar a seleção mais específica de dados e explorar diferentes hierarquias, o desempenho do modelo pode ser ainda mais otimizado, aumentando sua capacidade de fazer previsões precisas sobre a água disponível no solo.

A. Apêndice - Código Fonte da modelagem

Listing 1. Código do projeto

```
"""REGRESSOR_MD.ipynb
# Modelagem dados Hybras

## IMPORTS
"""

import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor,
```

```
RandomForestClassifier
from sklearn.svm import SVR
from xgboost import XGBRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.neural_network import MLPRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score,
mean_absolute_percentage_error, max_error
from sklearn.preprocessing import StandardScaler

"""## Fun  es

### Resumo boxplot and histograma
"""

def box_hist(df):

    fig, axs = plt.subplots(nrows=len(df.columns), ncols=2,
                            figsize=(6, 2 * len(df.columns)), dpi=300,)


    colors = {
        'limits': 'black',
        'box': 'darkblue',
        'fliers': 'red'
    }

    for i, col in enumerate(df.columns):

        bp = axs[i, 0].boxplot(df[col].values,labels=[col], vert=False,
                               notch = True,
                            patch_artist=True,

                            capprops = dict(color = "red", linewidth = 1),
                            whiskerprops = dict(color = "red", linewidth = 1),
                            boxprops =    dict(facecolor = "darkblue"),
                            medianprops = dict(color = "white",
                                               linewidth = 1.5)   )




        for flier in bp['fliers']:
            plt.setp(flier, markerfacecolor=colors['fliers'], marker='.',
                     markersize=8)

        for cap in bp['caps']:
            plt.setp(cap, color=colors['limits'], linewidth=2)

        for position in ['bottom', 'top', 'right', 'left']:
```

```
            axs[i,0].spines[position].set_color('black')
            axs[i,1].spines[position].set_color('black')



        axs[i, 0].set_xlim(0,  df[col].max()*1.1)
        axs[i, 1].grid(axis='x', linestyle='--', alpha=1, linewidth=1,
                       color='darkgray')
        axs[i, 0].grid(axis='x', linestyle='--', alpha=1, linewidth=1,
                       color='darkgray')

        if col in ['102', '15300']:
            axs[i, 0].set_yticklabels([f'{col} cm / cm '], rotation=90,
                                      verticalalignment='center')
            #Histograma
            axs[i, 1].hist(df[col], bins=12, alpha=1, label=f'{col} cm / cm ',
                           orientation='horizontal', color='darkblue')

        else:
            axs[i, 0].set_yticklabels([col], rotation=90,
                                      verticalalignment='center')

            #Histograma
            axs[i, 1].hist(df[col], bins=12, alpha=1, label=col,
                           orientation='horizontal', color='darkblue')

        axs[i, 1].legend()



    plt.tight_layout(pad=1.2)


    plt.show()

"""### OBSERVADO X ESTIMADO"""

def targetXestimativas(y_teste, Resultados_Test, legenda):
    plt.figure(figsize=(8, 26), dpi=300)
    x = range(len(y_teste))



    num_models = len(Resultados_Test)

    for i, nome in enumerate(Resultados_Test):
        plt.subplot( num_models, 1, i+1)
        value = Resultados_Test[nome][0][legenda]
        plt.plot(x, value, label=f'Valores Estimados')
```

```
        plt.plot(x, y_teste, label='Valores␣Observados')
        plt.title(f'Valores␣{nome}␣{legenda}␣vs.␣Valores␣Observados␣->␣R  :
␣␣␣␣␣␣␣␣␣{round(r2_score(y_teste,␣value),␣2)}')
        plt.xlabel('Amostras')
        plt.ylabel('theta␣(cm /cm )')
        plt.legend()

    plt.tight_layout(pad=2)

    plt.show()

"""### Testando os regressoes lineares"""

from functools import reduce
def coef_reg(model, estimados):

    if len(model.coef_) == len(list(estimados)):
        valores_coef = list(map(lambda x: x ,[ dado[0] * dado[1] for
                                              dado in list(zip(estimados,
                                                   model.coef_))]))
        return reduce(lambda x=0, y=0: x + y, valores_coef)

"""### Colunas Faltantes"""

def colunas_faltantes(dataframe, feature_coluna:str, qtd:int)->list:
    lista_nomes = []
    for indice, nome in enumerate(dataframe.index):
        if dataframe.iloc[indice][feature_coluna] > qtd:
            lista_nomes.append(nome)
    return lista_nomes

"""### Fonte De Dados

"""

load_data = lambda path: pd.read_excel(path)

"""### BoxPlot

"""

def boxplot(dados, titulo):
  # Cria  o do boxplot
  sns.set_theme(style="whitegrid")
  plt.boxplot(dados)

  # Remover r tulos do eixo x
  plt.gca().set_xticklabels([])
```

```
  # R tulo do eixo y
  plt.ylabel('%')

  # T tulo do gr fico
  plt.title(titulo)

  # Exibi  o do gr fico
  plt.show()

"""### Feature Selecion"""

def feature_sel_matrix(filtro, feature_referencia, correlation_matrix):
  correlated_features = correlation_matrix[abs(correlation_matrix[
      feature_referencia]) > filtro].index.tolist()
  list_of_correlated_features = [feature for feature in correlated_features
                                 if feature != feature_referencia]
  print(f"Caracter sticas correlacionadas com '{feature_referencia}':",
        list_of_correlated_features )
  return list_of_correlated_features

"""### Matriz"""

def matrix_corr(dados):
    correlation = dados.corr()

    # Personalizar estilo do Seaborn
    sns.set(font_scale=1.2)
    sns.axes_style('darkgrid')

    sns.set_style("whitegrid", {'axes.edgecolor': 'black',
                                'text.color': 'white'}
                                )

    plt.figure(figsize=(12, 8), dpi=300)

    plt.rcParams['font.weight'] = 'bold'
    plt.rcParams['axes.labelweight'] = 'bold'

    # Personalizar cores dos r tulos
    sns.heatmap(correlation, annot=True, cmap='coolwarm',
                fmt=".2f", linewidths=.5, annot_kws={"weight": "bold",
                                                     "color": "black"},
                )
    plt.show()

    return correlation

"""### Modelagem"""
```

```
def evaluate_regression_models(models, X_train, X_test, y_train, y_test):
    metrics = {
        'Model': [],
        'MAE': [],
        'MSE': [],
        'RMSE': [],
        'R2': [],
        'MAPE': [],
        'Max Error': []
    }

    for model in models:
        model.fit(X_train, y_train)
        y_pred = model.predict(X_test)
        mae = mean_absolute_error(y_test, y_pred)
        mse = mean_squared_error(y_test, y_pred)
        rmse = mean_squared_error(y_test, y_pred, squared=False)
        r2 = r2_score(y_test, y_pred)
        mape = mean_absolute_percentage_error(y_test, y_pred)
        m_error = max_error(y_test, y_pred)

        metrics['Model'].append(type(model).__name__)
        metrics['MAE'].append(mae)
        metrics['MSE'].append(mse)
        metrics['RMSE'].append(rmse)
        metrics['R2'].append(r2)
        metrics['MAPE'].append(mape)
        metrics['Max Error'].append(m_error)

    df_metrics = pd.DataFrame(metrics)
    return df_metrics

def cross_validation(models, X_train, X_test,y_train, y_test):

  performance = {
        'Model': [],
        'MAE': [],
        'MSE': [],
        'RMSE': [],
        'R2': [],
        'MAPE': [],
        'Params': {}
    }

  regression_eq = {

  }

  lista_regression = ['Linear Regressor', 'Ridge', 'Lasso']
```

```
  for model_name, (model, param_grid) in models.items():

          grid_search = GridSearchCV(estimator=model, param_grid=param_grid,
                                     cv=5, scoring='neg_mean_squared_error',
                                     n_jobs=-1)
          grid_search.fit(X_train, y_train)


          best_params = grid_search.best_params_

          best_model = grid_search.best_estimator_

          if (model_name in lista_regression):


            # best_model.fit(X_train, y_train)


            coefficients = best_model.coef_


            equation = f"{model_name}: "
            for i, coef in enumerate(coefficients):
                equation += f"({coef} * X{i}) + "
            equation += f"{best_model.intercept_}"


            regression_eq[model_name] = equation


          y_pred = best_model.predict(X_test)
          mae = mean_absolute_error(y_test, y_pred)
          mse = mean_squared_error(y_test, y_pred)
          rmse = mean_squared_error(y_test, y_pred, squared=False)
          r2 = r2_score(y_test, y_pred)
          m_err = mean_absolute_percentage_error(y_test, y_pred)
          print(f'{model_name}: {best_params}')
          performance['Model'].append(model_name)
          performance['MAE'].append(mae)
          performance['MSE'].append(mse)
          performance['RMSE'].append(rmse)
          performance['R2'].append(r2)
          performance['MAPE'].append(m_err)
          performance['Params'][model_name] = best_model

  return [performance, regression_eq]

"""### Graficos"""
```

```
def show_plt_metrics(models, X_train, X_test, y_train, y_test):
    # Avaliar os modelos de regress o
    results = evaluate_regression_models(models, X_train, X_test, y_train,
                                         y_test)
    sns.set_theme(style="whitegrid")
    # Plotar gr ficos separados para cada m trica
    plt.figure(figsize=(20, 18))

    # MAE
    plt.subplot(3, 2, 1)
    plt.bar(results['Model'], results['MAE'])
    plt.title('MAE')
    plt.xlabel('Modelos')
    plt.ylabel('Valor da M trica')
    plt.xticks(rotation=90)

    # MSE
    plt.subplot(3, 2, 2)
    plt.bar(results['Model'], results['MSE'])
    plt.title('MSE')
    plt.xlabel('Modelos')
    plt.ylabel('Valor da M trica')
    plt.xticks(rotation=90)

    # RMSE
    plt.subplot(3, 2, 3)
    plt.bar(results['Model'], results['RMSE'])
    plt.title('RMSE')
    plt.xlabel('Modelos')
    plt.ylabel('Valor da M trica')
    plt.xticks(rotation=90)

    # R
    plt.subplot(3, 2, 4)
    plt.bar(results['Model'], results['R2'])
    plt.title('R ')
    plt.xlabel('Modelos')
    plt.ylabel('Valor da M trica')
    plt.xticks(rotation=90)

    # MAPE
    plt.subplot(3, 2, 5)
    plt.bar(results['Model'], results['MAPE'])
    plt.title('MAPE')
    plt.xlabel('Modelos')
    plt.ylabel('Valor da M trica')
    plt.xticks(rotation=90)
```

```
    # Max Error
    plt.subplot(3, 2, 6)
    plt.bar(results['Model'], results['Max Error'])
    plt.title('Max Error')
    plt.xlabel('Modelos')
    plt.ylabel('Valor da M trica')
    plt.xticks(rotation=90)


    plt.tight_layout()
    plt.show()

def show_graphics(performance, pallete="deep"):
  plt.figure(figsize=(16, 6), dpi=300)
  sns.set_theme(style="whitegrid")
  sns.set_palette(pallete)
  # MAE
  plt.subplot(1, 5, 1)
  plt.bar(performance['Model'], performance['MAE'])
  plt.title('MAE')
  plt.xlabel('Modelos')
  plt.ylabel('Valor da M trica')
  plt.xticks(rotation=90)

  # RMSE
  plt.subplot(1, 5, 2)
  plt.bar(performance['Model'], performance['RMSE'])
  plt.title('RMSE')
  plt.xlabel('Modelos')
  plt.ylabel('Valor da M trica')
  plt.xticks(rotation=90)

  # R
  plt.subplot(1, 5, 3)
  plt.bar(performance['Model'], performance['R2'])
  plt.title(' R  ')
  plt.xlabel('Modelos')
  plt.ylabel('Valor da M trica')
  plt.xticks(rotation=90)

  # MAPE
  plt.subplot(1, 5, 4)
  plt.bar(performance['Model'], performance['MAPE'])
  plt.title('MAPE')
  plt.xlabel('Modelos')
  plt.ylabel('Valor da M trica')
  plt.xticks(rotation=90)

  plt.tight_layout()
```

```
  plt.show()

"""### Retornar CC, PMP e AD"""

def theta(modeloCC, modeloPMP, dados, X_train_CC, y_train_CC,
          X_train_PMP, y_train_PMP):

  model_cc = modeloCC
  model_pmp = modeloPMP

  model_cc.fit(X_train_CC, y_train_CC)
  model_pmp.fit(X_train_PMP, y_train_PMP)



  model_pred_cc = model_cc.predict(dados)
  model_pred_pmp = model_pmp.predict(dados)
  return [float(model_pred_cc), float(model_pred_pmp),float(model_pred_cc -
                                                            model_pred_pmp)*10 ]

def ensemble(modeloCC, modeloPMP, dados, X_train_CC, y_train_CC, X_train_PMP,
             y_train_PMP):
  cc = []
  pmp = []

  #INSTANCIA DO MELHOR MODELO
  model_cc = modeloCC
  model_pmp = modeloPMP

  #TREINAMENTO
  model_cc.fit(X_train_CC, y_train_CC)
  model_pmp.fit(X_train_PMP, y_train_PMP)
  #PREVER OS DADOS
  for row in range(len(dados)):
    cc.append(model_cc.predict(dados[row]))
    pmp.append(model_pmp.predict(dados[row]))
  return {'CC': cc, 'PMP': pmp}

"""### DUMP de modelos"""

def dump_models(modeloCC, modeloPMP, X_train_CC, y_train_CC, X_train_PMP,
                y_train_PMP):

  model_cc = modeloCC
  model_pmp = modeloPMP

  model_cc.fit(X_train_CC, y_train_CC)
  model_pmp.fit(X_train_PMP, y_train_PMP)
```

```
  return [model_cc, model_pmp]

"""### RFE
- Recursive Feature Elimination
"""

def RFE(df, coluna_alvo, random_state=42, test_size=0.2, n_features=5):

    X = df.drop(coluna_alvo, axis=1)
    y = df[coluna_alvo]

    X_train, X_test, y_train, y_test = train_test_split(X,
                                                        y, test_size=test_size,
                                                    random_state=random_state)

    rfc = RandomForestRegressor(random_state=random_state, n_estimators=100)

    rfc.fit(X_train, y_train)

    importancias = rfc.feature_importances_

    df_importancias = pd.DataFrame({'Feature': X.columns,
                                    'Importance': importancias})

    df_importancias = df_importancias.sort_values(by='Importance',
                                                  ascending=False)

    return df_importancias.head(n_features)

"""## Fonte de dados"""

df = load_data('./HYBRAS.xlsx')
df.head()

df.rename(columns={102:'102', 15300: '15300'}, inplace=True)

"""## EDA"""

result = pd.concat([df.mean().round(2),
                    df.median().round(2), df.std().round(2),
                    df.max().round(2),
                    df.min().round(2),round(df.isna().sum()/
                    df.isna().count(),2), df.dtypes, df.isna().sum()],
                    axis=1, keys=['M dia', 'Mediana', 'Desvio␣Padr o','M ximo',
                                  'M nimo', 'Dados␣Faltantes␣(%)',
                                  'Tipos', 'NaN␣Count' ])
result

colums = df.columns
```

```
valores_negativos = []
for i in colums:
    valores_negativos.append((i,df[df[i] < 0][i].count()))
valores_negativos = pd.DataFrame(valores_negativos, columns=['Features',
                                                             'Negative Count'])
valores_negativos

"""### Excluindo colunas com muitos dados faltantes"""

colunas_falt = colunas_faltantes(result, 'NaN Count', 100 )
colunas_falt

df = df.drop(columns=colunas_falt, axis=1)
df = df.dropna()

df.isna().sum()

df.drop(columns=['code'], inplace=True)

"""### Identificando Outliers

"""

df.describe()

#Garantindo que a granulometria nao esta ultrapando 100%
soma_fracoes = df[df.columns[0]] + df[df.columns[1]]  + df[df.columns[2]]

soma_fracoes[soma_fracoes > 100].describe()

granulometria = df[df.columns[:3]]
granulometria.describe()

"""### Limpeza de outliers
- A granulometria n o deve passar 100%, portanto identificou-se
que os parametros que est o passando est o com erro de unidade
"""

granulometria['Soma'] = granulometria.apply(lambda row: row['clay (%)']+
                                            row['silt (%)'] +
                                            row['sand (%)'] , axis=1)
granulometria[(df['clay (%)']>100) | (df['silt (%)'] > 100) |
              (df['sand (%)'] > 100) ]

df.describe()

data = df.copy()
data.dropna(inplace=True)
data.dtypes
```

```
"""### Boxplot e histograma das variaveis"""

data.describe()

df

box_hist(df[df.columns[3:]])

"""## Melhores Atributos

### Selecionar acima de 0.2

- Utilizando a matrix de correla  o para selecionar acima de 20%
"""

matrix_corr(df)

"""- Utilizando o RFE para selecionar os melhores hiperparametros"""

resultado_RFE = RFE(df.drop(columns='102'), '15300')
features_sel_RFE = resultado_RFE['Feature'][resultado_RFE['Importance']<0.1
                                            ].to_list()
features_sel_RFE.extend(['102', '15300', 'silt (%)'])
features_sel_RFE

"""## Separando os dados de treinamento e teste

- Para as duas variaveis alvos '102' e '15300'
"""

X_102 = df.drop(columns=['102','15300'], axis=1)
y_102 = df['102']

X_15300 = df.drop(columns=['102','15300'], axis=1)
y_15300 = df['15300']


X_train_102, X_test_102,
y_train_102, y_test_102 = train_test_split(X_102, y_102,
                                           test_size=0.2,random_state=42)
X_train_15300, X_test_15300,
 y_train_15300, y_test_15300 = train_test_split(X_15300, y_15300,
                                                test_size=0.2, random_state=42)

"""## Escolha do modelo

- Foram escolhidos 10 modelos
"""
```

```
model = [
    LinearRegression(),
    Ridge(),
    Lasso(),
    RandomForestRegressor(),
   GradientBoostingRegressor(),
    SVR(),
]

"""### Parametros do gridSearch"""

param_grid_random_forest = {
    'n_estimators': [100, 200, 300],
    'max_depth': [None, 5, 10],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

param_grid_svr = {
    'C': [0.1, 1, 10],
    'epsilon': [0.1, 0.01, 0.001]
}

param_grid_gradient_boosting = {
    'n_estimators': [100, 200, 300],
    'learning_rate': [0.1, 0.05, 0.01],
    'max_depth': [3, 5, 10]
}


param_grid_linear_regressor = {
    'fit_intercept': [True, False],
       'n_jobs': [1,10,15, 5000],
    'positive':[True, False],
  'copy_X': [True, False],

}

param_grid_ridge = {
    'alpha': [0.1, 1, 10,5000],
    'fit_intercept': [True, False],
    'copy_X': [True, False],
    'max_iter': [1000,5000,15000],

}

param_grid_lasso = {
    'alpha': [0.1, 1, 10],
```

```
    'fit_intercept': [True, False],
    'max_iter': [1000,5000,15000],
    'copy_X': [True, False],

}


models = {
   'Linear Regressor': (LinearRegression(), param_grid_linear_regressor),
  'Ridge': (Ridge(random_state=42), param_grid_ridge),
  'Lasso': (Lasso(random_state=42), param_grid_lasso),
  'Random Forest': (RandomForestRegressor(random_state=42),
                    param_grid_random_forest),
  'Gradient Boosting': (GradientBoostingRegressor(random_state=42),
                        param_grid_gradient_boosting),
  'SVR': (SVR(), param_grid_svr),

}

"""## Melhores Hiperparametros para o modelo

- Iremos separar utilizando agora somente com as melhores features
selecionadas utilizando a t cnica do RFE e Matrix
"""

features_sel_RFE

X_102 = df.drop(columns=['102','15300'], axis=1)
y_102 = df['102']

X_15300 = df.drop(columns=['102','15300'], axis=1)
y_15300 = df['15300']


X_train_102, X_test_102, y_train_102, y_test_102 = train_test_split(X_102,
 y_102, test_size=0.2, random_state=42)
X_train_15300, X_test_15300, y_train_15300,
 y_test_15300 = train_test_split(X_15300, y_15300, test_size=0.2,
                                 random_state=42)

"""## Valida  o cruzada

### CC 102
"""

performance_CC10 = cross_validation(models=models,
                                    X_train=X_train_102, X_test=X_test_102,
                                    y_train=y_train_102, y_test= y_test_102)
```

```
"""### PMP"""

performance_PMP = cross_validation(models=models, X_train=X_train_15300,
                                   X_test=X_test_15300, y_train=y_train_15300,
                                   y_test= y_test_15300)

"""### Tabela dos hiperparametros escolhidos para cada modelo"""

pd.DataFrame(list(zip([str(hiper) for nome,
                       hiper in performance_CC10[0]['Params'].items()],
                      [str(hiper) for nome,
                       hiper in performance_PMP[0]['Params'].items()])),
             columns=['CC', 'PMP'])

"""## Avalia  o

### CC10
"""

show_graphics(performance_CC10[0], 'dark')

"""### PMP

"""

show_graphics(performance_PMP[0], 'dark')

valores_observados = []

for index, row in X_test_102.iterrows():
    linha_array = np.array(row.values).reshape(1, -1)
    valores_observados.append(linha_array)

valores_observados[0].reshape(1, -1)

"""## Resultados

"""

Resultados_Test = {
    'Random Forest': [],
 #'Decision Tree': [],
 #'KNN Regressor': [],
 'SVR': [],
 'Gradient Boosting': [],
 #'XGBoost Regressor': [],
 'Linear Regressor': [],
 'Ridge': [],
 'Lasso': [],
```

```
 #'MLPRegressor':[]
}

for nome, key in models.items():

    Resultados_Test[nome].append(ensemble(performance_CC10[0]['Params'][nome],
                                          performance_PMP[0]['Params'][nome],
                                          valores_observados, X_train_102,
                                          y_train_102, X_train_15300,
                                          y_train_15300))

array_ensemble_CC= {}
array_ensemble_PMP= {}
tamanho_test = len(valores_observados)

for index in range(tamanho_test):
    array_ensemble_CC[index]=[]
    for nome, key in models.items():

      array_ensemble_CC[index].append(Resultados_Test[nome][0]['CC'][index])

for index in range(tamanho_test):
    array_ensemble_PMP[index]=[]
    for nome, key in models.items():

      array_ensemble_PMP[index].append(Resultados_Test[nome][0]['PMP'][index])

medias_cc = []
medias_pmp = []

for index in range(tamanho_test):
  medias_cc.append(np.mean(array_ensemble_CC[index]))
  medias_pmp.append(np.mean(array_ensemble_PMP[index]))

"""### Gr ficos do observado vs Estimados"""

metricas_cc_dataframe = pd.DataFrame(list(zip(performance_CC10[0]['Model'],
                                              performance_CC10[0]['RMSE'],
                                              performance_CC10[0]['R2'] )),
                                     columns=['Model', 'RMSE', 'R2'])
metricas_cc_dataframe

metricas_dataframe = pd.DataFrame(list(zip(performance_PMP[0]['Model'],
                                           performance_CC10[0]['RMSE'],
                                           performance_PMP[0]['RMSE'],
                                           performance_CC10[0]['R2'],
                                           performance_PMP[0]['R2'] )),
                                           columns=['Model', 'RMSE_CC',
                                            'RMSE_PMP', 'R2_CC','R2_PMP'])
```

```
metricas_dataframe

targetXestimativas(y_test_15300, Resultados_Test, "PMP")

targetXestimativas(y_test_102, Resultados_Test, "CC")

"""## Export Models"""

import joblib
import os

path = './models/'
models_folder = os.path.join(os.getcwd(), 'models')
if not os.path.exists(models_folder):
    os.makedirs(models_folder)

ModelsDump = {}
for nome, key in models.items():
    ModelsDump[nome]=dump_models(performance_CC10[0]['Params'][nome],
                                 performance_PMP[0]['Params'][nome],
                                 X_train_102, y_train_102, X_train_15300,
                                 y_train_15300)

for nome, key in models.items():
    cc_file_path = os.path.join(path, f'{nome}_cc.pkl')
    pmp_file_path = os.path.join(path, f'{nome}_pmp.pkl')
    joblib.dump(ModelsDump[nome][0], cc_file_path)
    joblib.dump(ModelsDump[nome][1], pmp_file_path)
```

Referências

BARROS, A. H. C. et al. Pedotransfer functions to estimate water retention parameters of soils in northeastern brazil. *Revista Brasileira de Ciência do Solo*, Sociedade Brasileira de Ciência do Solo, v. 37, p. 379–391, 4 2013. ISSN 0100-0683. Disponível em: ⟨http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0100-06832013000200009&lng=en&tlng=en⟩. 3, 10, 12, 13

BERRAR, D. Cross-validation. In: ____. Elsevier, 2018. v. 1, p. 542–545. ISBN 9780128096338. Disponível em: ⟨https://www.researchgate.net/publication/324701535_Cross-Validation⟩. 10

BOUMA, J. Using soil survey data for quantitative land evaluation. In: ____. *Advances in Soil Science: Volume 9*. New York, NY: Springer US, 1989. p. 177–213. ISBN 978-1-4612-3532-3. Disponível em: ⟨https://doi.org/10.1007/978-1-4612-3532-3_4⟩. 3

BOUYOUCOS, G. J. A recalibration of the hydrometer method for making mechanical analysis of soils1. *Agronomy Journal*, John Wiley Sons, Ltd, v. 43, p. 434–438, 9 1951. ISSN 1435-0645. Disponível em: ⟨https://onlinelibrary.wiley.com/doi/full/10.2134/agronj1951.00021962004300090005xhttps://onlinelibrary.wiley.com/doi/abs/10.2134/agronj1951.00021962004300090005xhttps://acsess.onlinelibrary.wiley.com/doi/10.2134/agronj1951.00021962004300090005x⟩. 3

BREIMAN, L. Random forests. *Machine Learning*, Springer, v. 45, p. 5–32, 10 2001. ISSN 08856125. Disponível em: ⟨https://link.springer.com/article/10.1023/A:1010933404324⟩. 11

CATLEY, C. et al. Extending crisp-dm to incorporate temporal data mining of multidimensional medical data streams: A neonatal intensive care unit case study. In: *2009 22nd IEEE International Symposium on Computer-Based Medical Systems*. [S.l.: s.n.], 2009. p. 1–5. 3, 4

CHAI, T.; DRAXLER, R. R. Root mean square error (rmse) or mean absolute error (mae)? – arguments against avoiding rmse in the literature. *Geoscientific Model Development*, Copernicus GmbH, v. 7, p. 1247–1250, 6 2014. ISSN 1991-9603. Disponível em: ⟨https://gmd.copernicus.org/articles/7/1247/2014/⟩. 14

CHICCO, D.; WARRENS, M. J.; JURMAN, G. The coefficient of determination r-squared is more informative than smape, mae, mape, mse and rmse in regression analysis evaluation. *PeerJ Computer Science*, v. 7, p. e623, jul. 2021. ISSN 2376-5992. Disponível em: ⟨https://doi.org/10.7717/peerj-cs.623⟩. 15

COIMBRA, J. L. M. et al. Conseqüências da multicolinearidade sobre a análise de trilha em canola. *Ciência Rural*, Universidade Federal de Santa Maria, v. 35, p. 347–352, 4 2005. ISSN 0103-8478. Disponível em: ⟨http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0103-84782005000200015&lng=pt&tlng=pt⟩. 9

CORTéS, U. et al. Artificial intelligence and environmental decision support systems. *Appl. Intell.*, v. 13, p. 77–91, 6 2000. Disponível em: ⟨http://dx.doi.org/10.1023/A:1008331413864⟩. 2

DONAGEMMA, G. K. et al. *Manual de métodos de análise de solo*. 2nd rev. ed.. ed. Rio de Janeiro: Embrapa Solos, 2011. 230 p. (Embrapa Solos. Documentos, 132). Biblioteca(s): Embrapa Solos. ISSN 1517-2627. Disponível em: ⟨https://ainfo.cnptia.embrapa.br/digital/bitstream/item/104933/1/Manual-de-Mtodos-de-Anilise-de-Solo.pdf⟩. 7, 17

FILHO., T. B. O.; CAETANO., A. R.; OTTONI., M. V. In situ field capacity in brazilian soils and a derived irrigation management practice based on water suction. *Journal of Agricultural Science*, v. 14, n. 3, p. 17, 2022. 2

GUNARATHNA, M. H. et al. Machine learning approaches to develop pedotransfer functions for tropical sri lankan soils. *Water 2019, Vol. 11, Page 1940*, Multidisciplinary Digital Publishing Institute, v. 11, p. 1940, 9 2019. ISSN 2073-4441. Disponível em: ⟨https://www.mdpi.com/2073-4441/11/9/1940/htmhttps://www.mdpi.com/2073-4441/11/9/1940⟩. 3, 4

HEMPHILL, J. F. Interpreting the magnitudes of correlation coefficients. *American Psychologist*, v. 58, p. 78–79, 1 2003. ISSN 1935-990X. Disponível em: ⟨http://doi.apa.org/getdoi.cfm?doi=10.1037/0003-066X.58.1.78⟩. 15

HILLEL, D. *Introduction to Environmental Soil Physics*. Elsevier, 2003. 1-494 p. ISBN 9780123486554. Disponível em: ⟨https://linkinghub.elsevier.com/retrieve/pii/B9780123486554X5000X⟩. 7

HODNETT, M. G.; TOMASELLA, J. Marked differences between van genuchten soil water-retention parameters for temperate and tropical soils: A new water-retention pedo-transfer functions developed for tropical soils. *Geoderma*, v. 108, p. 155–180, 2002. ISSN 00167061. Disponível em: ⟨https://www.researchgate.net/publication/222514532_Marked_differences_between_van_Genuchten_soil_water-retention_parameters_for_temperate_and_tropical_soils_A_new_water-retention_pedo-transfer_functions_developed_for_tropical_soils⟩. 2, 5, 10, 12

LAVALLE, S. M.; BRANICKY, M. S.; LINDEMANN, S. R. On the relationship between classical grid search and probabilistic roadmaps. *The International Journal of Robotics Research*, v. 23, n. 7-8, p. 673–692, 2004. Disponível em: ⟨https://doi.org/10.1177/0278364904045481⟩. 11

MORAES, S.; LIBARDI, P.; NETO, D. D. Problemas metodológicos na obtenção da curva de retenção da água pelo solo. *Scientia Agricola*, FapUNIFESP (SciELO), v. 50, p. 383–392, 12 1993. Disponível em: ⟨https://www.researchgate.net/publication/250043217_Problemas_metodologicos_na_obtencao_da_curva_de_retencao_da_agua_pelo_solo⟩. 2

MORENO, J. J. M. et al. Using the r-mape index as a resistant measure of forecast accuracy. *Psicothema*, Psicothema, v. 25, p. 500–6, 2013. ISSN 1886-144X. Disponível em: ⟨http://www.ncbi.nlm.nih.gov/pubmed/24124784⟩. 14

MYTTENAERE, A. et al. Mean absolute percentage error for regression models. *Neurocomputing*, v. 192, p. 38–48, 6 2016. ISSN 09252312. Disponível em: ⟨https://linkinghub.elsevier.com/retrieve/pii/S0925231216003325⟩. 14

OLIVEIRA, L. B. et al. Funções de pedotransferência para predição da umidade retida a potenciais específicos em solos do estado de pernambuco. *Revista Brasileira de Ciência do Solo*, Sociedade Brasileira de Ciência do Solo, v. 26, p. 315–323, 6 2002. ISSN 0100-0683. Disponível em: ⟨http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0100-06832002000200004&lng=pt&tlng=pt⟩. 2, 3, 10, 13, 23

OTTONI, M. V. et al. Hydrophysical database for brazilian soils (hybras) and pedotransfer functions for water retention. *Vadose Zone Journal*, John Wiley Sons, Ltd, v. 17, p. 1–17, 1 2018. ISSN 1539-1663. Disponível em: ⟨https://onlinelibrary.wiley.com/doi/full/10.2136/vzj2017.05.0095https://onlinelibrary.wiley.com/doi/abs/10.2136/vzj2017.05.0095https://acsess.onlinelibrary.wiley.com/doi/10.2136/vzj2017.05.0095⟩. 5, 6

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011. 10, 13

PEREIRA, T. dos S. et al. The use of artificial intelligence for estimating soil resistance to penetration. *Engenharia Agrícola*, Associação Brasileira de Engenharia Agrícola, v. 38, p. 142–148, 1 2018. ISSN 1809-4430. Disponível em: ⟨http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0100-69162018000100142&lng=en&tlng=en⟩. 3

PODGORELEC, V. et al. Decision trees: An overview and their use in medicine. *Journal of Medical Systems*, Springer, v. 26, p. 445–463, 10 2002. ISSN 01485598. Disponível em: ⟨https://link.springer.com/article/10.1023/A:1016409317640⟩. 11

RAMCHARAN, A. et al. A soil bulk density pedotransfer function based on machine learning: A case study with the ncss soil characterization database. *Soil Science Society of America Journal*, John Wiley Sons, Ltd, v. 81, p. 1279–1287, 11 2017. ISSN 1435-0661. Disponível em: ⟨https://onlinelibrary.wiley.com/doi/full/10.2136/sssaj2016.12.0421https://onlinelibrary.wiley.com/doi/abs/10.2136/sssaj2016.12.0421https://acsess.onlinelibrary.wiley.com/doi/10.2136/sssaj2016.12.0421⟩. 3, 4

SAHOUR, H. et al. Random forest and extreme gradient boosting algorithms for streamflow modeling using vessel features and tree-rings. *Environmental Earth Sciences*, Springer Science and Business Media Deutschland GmbH, v. 80, 11 2021. ISSN 18666299. Disponível em: ⟨https://www.researchgate.net/publication/355828449_Random_forest_and_extreme_gradient_boosting_algorithms_for_streamflow_modeling_using_vessel_features_and_tree-rings⟩. 11

SANTOS, H. G. et al. *Sistema Brasileiro de Classificação de Solos*. 5th. ed. Brasília, DF: Embrapa, 2018. 356 p. Il. color. ; 16 cm x 23 cm. ISBN 978-85-7035-800-4. 6, 7, 17

SAXTON, K. E. et al. Estimating generalized soil-water characteristics from texture. *Soil Science Society of America Journal*, Wiley, v. 50, p. 1031–1036, 7 1986. ISSN 0361-5995. Disponível em: ⟨https://acsess.onlinelibrary.wiley.com/doi/10.2136/sssaj1986.03615995005000040039x⟩. 3

SCHRöER, C.; KRUSE, F.; GóMEZ, J. M. A systematic literature review on applying crisp-dm process model. *Procedia Computer Science*, v. 181, p. 526–534, 2021. ISSN 1877-0509. CENTERIS 2020 - International Conference on ENTERprise Information Systems / ProjMAN 2020 - International Conference on Project MANagement / HCist 2020 - International Conference on Health and Social Care Information Systems and Technologies 2020, CENTERIS/ProjMAN/HCist 2020. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S1877050921002416⟩. 4

SEDAGHAT, A. et al. Developing pedotransfer functions using sentinel-2 satellite spectral indices and machine learning for estimating the surface soil moisture. *Journal of Hydrology*, Elsevier, v. 606, p. 127423, 3 2022. ISSN 0022-1694. 3, 4

SHRESTHA, N. Detecting multicollinearity in regression analysis. *American Journal of Applied Mathematics and Statistics*, Science and Education Publishing Co., Ltd., v. 8, p. 39–42, 6 2020. ISSN 2328-7306. Disponível em: ⟨http://pubs.sciepub.com/ajams/8/2/1/index.html⟩. 9, 12, 13

SILVA, P. d. et al. Funções de pedotransferência para as curvas de retenção de água e de resistência do solo à penetração. *Revista Brasileira de Ciência do Solo*, Sociedade Brasileira de Ciência do Solo, v. 32, p. 1–10, 2 2008. ISSN 0100-0683. Disponível em: ⟨http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0100-06832008000100001&lng=pt&tlng=pt⟩. 2, 10, 12

SU, X.; YAN, X.; TSAI, C. L. Linear regression. *Wiley Interdisciplinary Reviews: Computational Statistics*, John Wiley Sons, Ltd, v. 4, p. 275–294, 5 2012. ISSN 1939-0068. Disponível em: ⟨https://onlinelibrary.wiley.com/doi/full/10.1002/wics.1198https://onlinelibrary.wiley.com/doi/abs/10.1002/wics.1198https://wires.onlinelibrary.wiley.com/doi/10.1002/wics.1198⟩. 12

SUTTER, J.; KALIVAS, J. Comparison of forward selection, backward elimination, and generalized simulated annealing for variable selection. *Microchemical Journal*, Elsevier, v. 47, p. 60–66, 2 1993. ISSN 0026265X. Disponível em: ⟨https://linkinghub.elsevier.com/retrieve/pii/S0026265X8371012X⟩. 13

TOMASELLA, J.; HODNETT, M. G.; ROSSATO, L. Pedotransfer functions for the estimation of soil water retention in brazilian soils. *Soil Science Society of America Journal*, v. 64, p. 327–338, 1 2000. ISSN 0361-5995. Disponível em: ⟨https://acsess.onlinelibrary.wiley.com/doi/10.2136/sssaj2000.641327x⟩. 2, 5, 8, 10, 12, 13

XIA, Z. et al. Application of genetic algorithmsupport vector regression model to predict damping of cantilever beam with particle damper. *Journal of Low Frequency Noise Vibration and Active Control*, SAGE Publications Inc., v. 36, p. 138–147, 6 2017. ISSN 20484046. Disponível em: ⟨https://journals.sagepub.com/doi/10.1177/0263092317711987⟩. 12

YE, Z. et al. Tackling environmental challenges in pollution controls using artificial intelligence: A review. *Science of The Total Environment*, v. 699, p. 134279, 2020. ISSN 0048-9697. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0048969719342627⟩. 2

YOO, W. et al. A study of effects of multicollinearity in the multivariable analysis. *International journal of applied science and technology*, NIH Public Access, v. 4, p. 9–19, 10 2014.

ISSN 2221-0997. Disponível em: ⟨http://www.ncbi.nlm.nih.gov/pubmed/25664257http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4318006⟩. 9